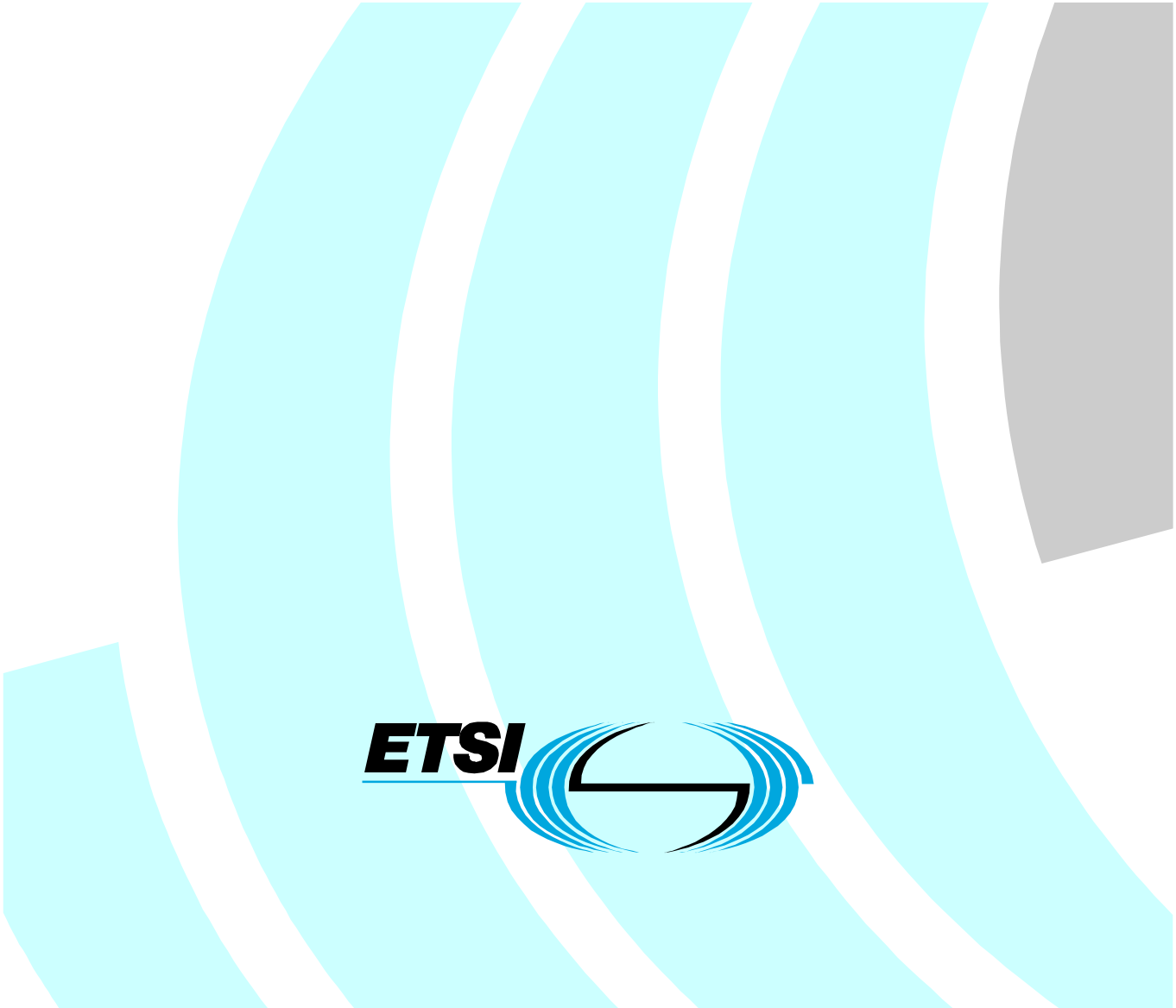
ETSI ES 202 912-5 V1.1.1 (2003-02)

ETSI Standard

Access and Terminals (AT);
Short Message Service (SMS) for PSTN/ISDN
Test Suites for SMS User Based Solution;
Part 5: Test Suite Structure and Test Purposes (TSS&TP)
user side for Data Link Layer (DLL) Protocol 2



Reference
DES/AT-030014-05

Keywords
SMS, ISDN, PSTN, TSS&TP

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, send your comment to:
editor@etsi.org

Copyright Notification

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

All published ETSI deliverables shall include information which directs the reader to the above source of information.

Foreword

This ETSI Standard (ES) has been produced by ETSI Technical Committee Access and Terminals (AT).

The present document is part 5 of a multi-part deliverable. Full details of the entire series can be found in part 1 [10].

1 Scope

The present document provides test suite structure and test purposes for testing Data Link layer in a Terminal Equipment implementing the Short Message Service (SMS) for PSTN/ISDN, UBS Protocol 2 according to ES 201 912 [1].

Basic ISDN or PSTN call procedures apply in order to establish a circuit-switched band connection between such Terminal Equipment and an SM-SC. Tests for these procedures are outside the scope of the present document. UBS2 terminals send and receive Data Link messages in the voice-band connection using the FSK signalling as defined in EN 300 659-2 [3] and ES 200 778-2 [6]. Tests for the FSK signalling are also outside the scope of the present document.

Terminal Equipment implementing the Short Message Service (SMS) for PSTN/ISDN according to UBS Protocol 2 are required to implement the Transfer Layer according to ES 201 912 [1]. Using the Remote Single Layer Embedded Test Method (see ISO/IEC 9646-2 [9]) for the UBS Protocol 2 Data Link layer , Transfer Layer messages appear here only as octet string parameters of Data Link layer messages. Tests for the Transfer Layer are not within the scope of the present document.

Figure 1 gives an overview of the reference architecture used for the UBS Protocol 2 operation. Figure 2 shows the configuration used for testing.

ISO/IEC 9646-1 [8] and ISO/IEC 9646-2 [9] are used as the basis for the test specification methodology.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication and/or edition number or version number) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

[1] ETSI ES 201 912 (V1.1.1): "Access and Terminals (AT); Short Message Service (SMS) for PSTN/ISDN; Short Message Communication between a fixed network Short Message Terminal Equipment and a Short Message Service Centre".

[2] ETSI EN 300 659-1 (V1.3.1): "Access and Terminals (AT); Analogue access to the Public Switched Telephone Network (PSTN); Subscriber line protocol over the local loop for display (and related) services; Part 1: On-hook data transmission".

[3] ETSI EN 300 659-2 (V1.3.1): "Access and Terminals (AT); Analogue access to the Public Switched Telephone Network (PSTN); Subscriber line protocol over the local loop for display (and related) services; Part 2: Off-hook data transmission".

[4] ETSI ES 200 659-3 (V1.3.1): "Access and Terminals (AT); Analogue access to the Public Switched Telephone Network (PSTN); Subscriber line protocol over the local loop for display (and related) services; Part 3: Data link message and parameter codings".

[5] ETSI ES 200 778-1 (V1.3.1): "Access and Terminals (AT); Analogue access to the Public Switched Telephone Network (PSTN); Protocol over the local loop for display and related services; Terminal equipment requirements; Part 1: On-hook data transmission".

[6] ETSI ES 200 778-2 (V1.3.1): "Access and Terminals (AT); Analogue access to the Public Switched Telephone Network (PSTN); Protocol over the local loop for display and related services; Terminal equipment requirements; Part 2: Off-hook data transmission".

[7] ETSI ES 202 912-4 (V1.1.1): "Access and Terminals (AT); Short Message Service (SMS) for PSTN/ISDN Test Suites for SMS User Based Solution; Part 4: Protocol Implementation Conformance Statement (PICS) proforma specification user side for Data Link Layer Protocol 2".

[8] ISO/IEC 9646-1: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 1: General concepts".

[9] ISO/IEC 9646-2: "Information technology - Open systems interconnection - Conformance testing methodology and framework - Part 2: Abstract test suite specification".

[10] ETSI ES 202 912-1 (V1.1.1): "Access and Terminals (AT); Short Message Service (SMS) for PSTN/ISDN; Test Suites for SMS User Based Solution; Part 1: Protocol Implementation Conformance Statement (PICS) proforma specification user side for Data Link Layer (DLL) Protocol 1".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

DL-Initiator: entity (SM-TE or SM-SC) initiating a data link connection to the peer entity by sending a DL establishing message (after the VB-connection has been established)

DL-Responder: entity (SM-TE or SM-SC) having received a DL establishing message from the peer entity (after the VB-connection has been established)

incoming VB-connection: VB-connection initiated by an SM-SC

inopportune behaviour (of the tester): The tester sends a message which is not expected by the IUT under the current circumstances (state).

invalid behaviour (of the tester): abbreviated form of "syntactically invalid behaviour", i.e. the tester sends a message where the presence or contents of one or more parameters or fields does not conform to the requirements

Lower Test System (LTS): part of the Test System performing basic signalling procedures to establish a VB-connection and to transmit and receive FSK frames

originator (of an SM): SM-TE sending an SM to another SM-TE

outgoing VB-connection: VB-connection initiated by an SM-TE

peer (entities): SM-TE and SM-SC, for which a voice-band connection exists or is pending, are considered as peers

valid behaviour (tests): The tester behaves according to the protocol.

NOTE: Timeout tests normally belong to the valid tests.

VB-connection: voice-band connection between two peers, considered to be **completed** or **established**, when the basic call control procedures, performed according to the type of network access the SM-TE is connected to (i.e. PSTN or BRA ISDN or PRA ISDN), are completed and the voice-band connection is ready for FSK frame transfer

NOTE: In order to establish an incoming VB-connection, the CLI information has to be previously provided to the terminal equipment. The way of providing CLI (e.g. by DTMF or FSK signalling and using a data transmission associated with ringing or not, etc., in the case of PSTN) is out of the scope of the present document.

VB-Initiator: entity (SM-TE or SM-SC) initiating a voice-band connection to the peer entity

VB-Responder: entity (SM-TE or SM-SC) having received a voice-band connection attempt from the peer entity

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ASP	Abstract Service Primitive
ATS	Abstract Test Suite
CLI	Calling Line Identification (information)
CM	Connection Manager
DL	Data Link
DLL	Data Link Layer
DTMF	Dual Tone Multi-Frequency
FSK	Frequency Shift Keying
INOP	Subgroup extension for "Inopportune Behaviour" tests
INV	Subgroup extension for "Invalid Behaviour" tests
ISDN	Integrated Services Digital Network
ISO	International Standard Organization
IUT	Implementation Under Test
LT	Lower Tester
PDU	Protocol Data Unit
PICS	Protocol Implementation Conformance Statement
PIXIT	Protocol Implementation eXtra Information for Testing
PSTN	Public Switched Telephone Network
SM	Short Message(s)
SME	Short Message Entity
SMS	Short Message Service
SM-SC	Short Message Service Centre
SM-TE	Short Message Terminal Equipment
SM-TL	Short Message Transfer Layer
SUT	System Under Test
TL	Transfer Layer
TP	Test Purpose
TSS	Test Suite Structure
TSS&TP	Test Suite Structure and Test Purposes
TTCN	Tree and Tabular Combined Notation
UBS	User Based Solution
UT	Upper Tester
VAL	Subgroup extension for "Valid Behaviour" tests
VB	Voice-band

4 Configuration assumed for the test specification

Figure 2 in clause 4 of ES 201 912 [1] shows the general principle of short message transfer, which consist in a SM submission phase and a SM delivery phase, which is repeated schematically in figure 1.

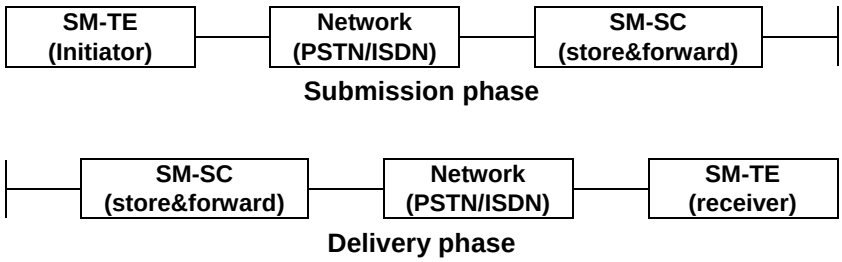


Figure 1: Short Message transfer - General Principle

In the tests specified in the present document, the SM-TE will be tested as an SM originator as well as a SM receiver. No different test configurations are defined for the two different roles the SM-TE take in the two phases. The test configuration shown in figure 2 therefore reflects each of the two phases of figure 1, with some necessary additions.

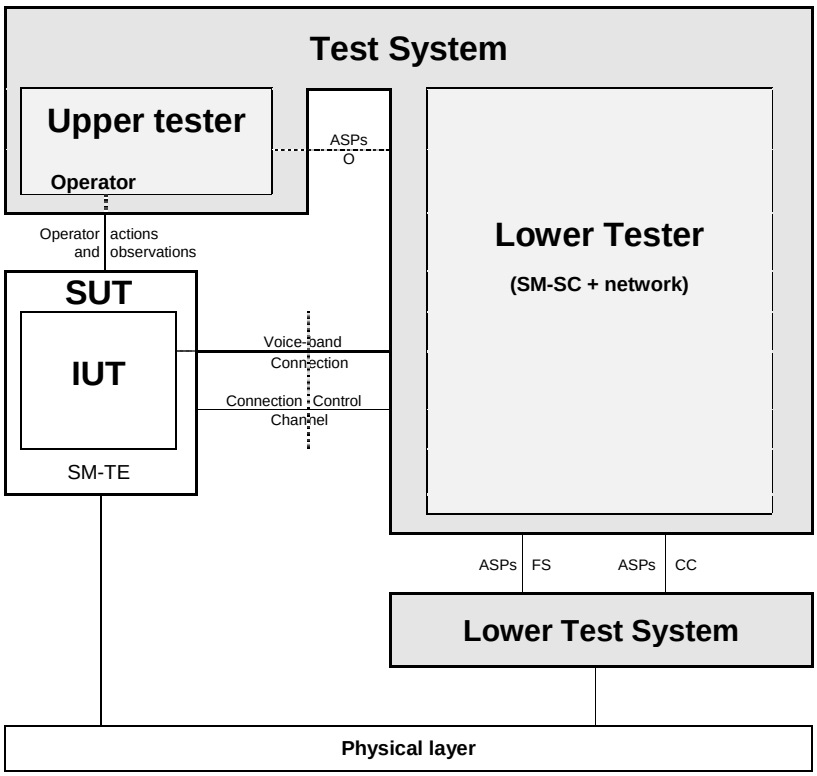


Figure 2: UBS2 test configuration

Explanations:

1) The SUT is an SM-TE.

2) The Test Method used here is the "Remote Single Layer Embedded" Test Method (see ISO/IEC 9646-1 [8]).

3) The IUT is the Data Link Layer implementation of the SM-TE.

4) The test system represents the network to which the IUT is attached, and the SM-SC. Correspondingly the SM-TE is physically connected to the test system as if it were connected to a real network.

5) The test system carries an Upper Tester and a Lower Tester. The Upper Tester is realized by an operator, controlling and observing the SUT. The test coordination between Lower Tester and the Upper Tester is performed via ASPs at PCO O.

6) Conceptually IUT and Lower Tester are connected by two channels:

   a) the voice-band channel, and

   b) the signalling control channel.

7) Virtually the Lower Tester communicates with the IUT via two PCOs:

   a) **FS**
   At this PCO the Lower Tester sends and receives FSK frames conceptually exchanged between SM-TE and SM-SC (which are transported over the voice-band channel).

b) **CC**
At this PCO the Lower Tester takes the actions of initiating and supervising connection control. The actions are realized by appropriate ASPs (see clause 6.3), while the actual transfer of signalling messages over the CC channel is not subject of this test specification.

Although ringing signals are physically transported over the voice-band connection, they are conceptually associated to the CC channel in this testing scheme. This is done to keep the **FS** PCO free from any signals except for FSK frames conceptually exchanged between SM-TE and SM-SC.

The same principle applies to the transfer of CLI: in the case where CLI has to be transported via FSK signalling, it is done via PCO CC. It is a matter for the test system to distinguish between these different ways of using FSK signalling.

5 Test purposes development

5.1 Introduction

A TP is defined for one or several conformance requirements to be tested. It is expected, that each TP will result in a test case keeping the same name, specified in an ATS based on the present document.

5.2 Source of test purpose specifications

The test purposes are based on the requirements made in ES 201 912 [1].

Requirements can be classified in "static" requirements and "dynamic" requirements. In the test purpose description tables of the present document, only the "dynamic" requirements of ES 201 912 [1] are referred to.

NOTE: The conformance requirements of ES 201 912 [1], "static" and "dynamic", are collected in ES 202 912-4 [7] (UBS2 PICS), but the PICS tables items are not referred to in the current document.

5.3 Restrictions and requirements not being tested

ES 201 912 [1] contains requirements for SM-TEs by direct specification and by reference to other standards. In particular, ES 201 912 [1] refers to the standards dealing with FSK signalling: EN 300 659-1 [2], EN 300 659-2 [3], EN 300 659-3 [4], ES 200 778-1 [5] and ES 200 778-2 [6].

It is not the intention of the present document to test Physical Layer aspects of FSK signalling, therefore there are no explicit tests except from those requirements related to FSK which are explicitly stated in ES 201 912 [1] (frame synchronization and delimitation, time interval requirements for frames).

According to ES 201 912 [1] SM-TEs are attached to PSTN or ISDN networks and perform basic call control procedures according to the type of the network access to establish a voice-band connection between the SM-TE and the SM-SC.

It is not within the scope of the present document to test any signalling associated with basic call control procedures. It is a matter of the test system implementing these tests to install and perform the appropriate procedures.

The initiation and supervision of these procedures is realized in the present document by specific ASPs (see clause 6.3).

According to ES 201 912 [1] the network, i.e. the tester implementing the tests specified here, has to provide the CLI of the SM-SC to the called SM-TE. When the network is a PSTN, CLI can be provided by DTMF signalling or by FSK signalling. It is also a matter of the test system to implement and use the method appropriate for the SM-TE and the network access type.

Only requirements that can be verified by inspection of the UBS Protocol 2 Data Link layer messages transferred over the voice-band channel are related to test purposes, in coordination with the ASPs necessary for terminal control via an operator and ASPs controlling voice-band connections, CLI provision and ringing.

TL contents of DLL messages are provided as Test Parameters of octet string type, i.e. it is a matter of parameterization of the executable test suite to provide valid TL contents for testing the DLL.

5.4 Grouping of test purposes

ES 201 912 [1] specifies requirements for frame synchronization, delimitation and time intervals between frames, as well as connection procedures at the DLL, which may be initiated by either side. The tests are grouped therefore in two main categories:

1) FSK Frame transfer related tests, and

2) DLL-connection-related tests.

The DLL-connection-related tests have been grouped into those for outgoing and for incoming connections. Within these groups, subgroups have been defined for the connection phases "establishment", "data transfer" and "release".

For each such subgroup "valid" tests have been specified. "Invalid" and "inopportune" tests have also been specified when appropriate.

Detailed information on the test groups and subgroups can be found in clause 6.2.

5.5 Test purpose naming convention

The current document is created together with other similar TSS&TP documents, dealing also with UBS Protocol 1 (UBS1) instead of UBS Protocol 2 (UBS2) and TL instead of DLL (see ES 201 912 [1]).

Therefore the Test purpose naming convention refers also to protocol and layer.

TP names are composed as follows:

Identifier := <Protocol>_<Layer>_<group structure>_<nn>

Where:

<Protocol> = **UBS2**, <Layer> = **DLL** and <nn> is a 2-digit sequential number, starting from **01** for each subgroup.

<group structure> is composed of sub-identifiers according to the subgroup structure of the group.

Example:

UBS2_DLL_OUT_EST_VAL_01

In this case **OUT** refers to group "Outgoing call", **EST** represents subgroup "establishment" and **VAL** refers to subgroup "Valid tests".

For details on groups and subgroups see table 1 in clause 6.2.

5.6 Method used for developing test purposes

Test purposes have only been developed for behaviour of the IUT which is visible at the FSK signalling interface. Following this basic principle, the development of the test purposes has been performed according to the following actions:

1) Using the PICS in ES 202 912-4 [7], identify all clauses of ES 201 912 [1] containing "dynamics" requirements and identify these requirements.

2) Determine the requirements not to be tested and identify them (see clause 5.3).

3) Define the test purposes structure (see clause 6.2).

4) Regarding that the structure is based on the DLL procedures defined in ES 201 912 [1] and that valid, invalid and inopportune test purposes have to be defined: identify the general requirements on invalid or inopportune messages and take them into account for all procedures, as far as applicable.

5) The MSCs in Annex B of ES 201 912 [1] are taken into account.

6) The method chosen for the presentation of the test purposes is described in clause 5.7. The method should enable an easy and systematic transition from the test purposes to a TTCN ATS.

7) When all test purposes have been defined, the "requirements references" are collected and checked with the list of all testable requirements, to ensure a suitable coverage.

5.7 Method used for test purpose description

A TP is described using a table as shown in the following example. The item names appearing in the left column of the example table are present in each TP table.

The right column of the example table contains *descriptive text* and example entries. The descriptive text is in italics, while example entries are in **bold text**.

The rows "Purpose", "Requirem. Ref." and "Selection Expr." contain normative contents, the rows "Preamble", "Test description", "Pass criteria" and "Postamble" contain just informative contents (the same information, expressed in the TTCN formalism, appear as normative in the .GR file related to the ATS document correspondent to the present document).

	Test Purpose identifier like **CON_OUT_EST_VB_01**
Purpose:	*<Textual description of the purpose to be achieved>*
Requirem. ref.:	*Reference to one or more clauses of ES 201 912 [1], containing a requirement which is tested in connection with the current test purpose. References to standards other than ES 201 912 [1] are added if appropriate. Since ES 201 912 [1] is the basic standard on which this test specification relies, the first paragraph of the "Requirem. ref." cell contains only clause numbers, without identifying the standard explicitly. Whenever a reference to another standards are necessary, a new paragraph is added, identifying the standard, followed by a colon and then clause numbers as for ES 201 912 [1].* **Example:** 6.3.2.1, table 5, B.1.4.1 EN 300 659-2 [3], clause 7.3.1
Selection Expr.:	*When an entry is present here, it denotes a selection expression (see clause 6.7), specifying a* ***condition for the applicability*** *of this test purpose. If there is no entry, the test purpose is unconditionally applicable.* *Note that selection expressions can also be defined for a* ***whole group of test purposes****, i.e. outside a test purpose table.*
Preamble:	*Name of a preamble leading to a condition or state, where the test purpose can be verified.*
Test description:	*Sequence of events intending to lead to the verification of the test purpose.* *A TTCN-like notation is used (see clause 6.6).* *Note that unexpected events are not shown here.*
Pass criteria:	*Special indication of an event (or several events), an "ignore"-behaviour or specific received parameter value(s), being essential for the verification of the test purpose.* *This can be a copy of one or more lines of the "Test description", or can be a textual explanation.*
Postamble:	**None** *or name of a postamble leading to the IDLE condition of the IUT (no VB connection exists).*

6 Test purposes presentation and environment specification

6.1 Introduction

After the definition of the **Test Suite Structure**, a suitable environment has to be created in order to formulate the test purpose descriptions properly (i.e. referring to elements of this environment). Apart from this, the environment is specified to support a **systematic transition** from the test purposes to the TTCN ATS. This is e.g. taken into account in the naming conventions and "atomic phrases" used in the descriptions. "**Test parameters**" and other objects have been collected during the test purposes development, which will find their entry in the ATS, with few modifications, e.g. as **TS Parameters**.

6.2 Test Suite Structure (TSS)

Table 1 shows the structure of the UBS2 Data Link Layer Test Suite, as well as the TP group identifiers and the number of test purposes produced, per subgroup and in total.

Table 1: Test suite structure for testing the UBS2 Data Link layer behaviour

Procedure/Group	Subgroup	Subgroup	Group identifier	Count
Frame transfer	Synchronization	Valid	UBS2_DLL_FRM_SYNC_VAL	6
	Timing intervals	Valid	UBS2_DLL_FRM_TIM_VAL	25
Outgoing call	Establishment	Valid	UBS2_DLL_OUT_EST_VAL	2
	Data transfer	Valid	UBS2_DLL_OUT_DAT_VAL	23
		Invalid	UBS2_DLL_OUT_DAT_INV	7
		Inopportune	UBS2_DLL_OUT_DAT_INOP	4
	Release	Valid	UBS2_DLL_OUT_REL_VAL	13
		Invalid	UBS2_DLL_OUT_REL_INV	6
		Inopportune	UBS2_DLL_OUT_REL_INOP	1
Incoming call	Establishment	Valid	UBS2_DLL_INC_EST_VAL	1
	Data transfer	Valid	UBS2_DLL_INC_DAT_VAL	16
		Invalid	UBS2_DLL_INC_DAT_INV	7
	Release	Valid	UBS2_DLL_INC_REL_VAL	7
		Invalid	UBS2_DLL_INC_REL_INV	3
Total:				**121**

6.3 Abstract Service Primitives

Three classes of ASPs are defined, according to the **PCO**s at which they operate (see figure 2):

PCO	ASP class
O:	ASPs related to the operator (Upper Tester),
FS:	ASPs for SM-related to the transmission and reception of FSK frames, and
CC:	ASPs related to connection control.

Table 2 describes the general functions of the ASPs operating at the 3 PCOs. Detailed descriptions of the ASPs together with their parameters follow.

Table 2: List of ASPs

PCO	ASP Name	Direction	Description
O	OUTGOING_CALL_req	LT->UT	Make the SM-TE initiate a VB connection to the SM-SC in order to send an SM.
	SET_MEM_FULL_req	LT->UT	Request the user to make the SM-TE memory full.
	SET_MEM_FULL_ind	UT->LT	Indicates that the SM-TE is in the Memory Full state.
	EMPTY_MEM_req	LT->UT	Request the user to delete one or more SMs to make the SM-TE memory again available.
	EMPTY_MEM_ind	UT->LT	Indicates that the SM-TE is not anymore in the Memory Full state.
CC	INC_CALL_req	LT->LTS	Make the Lower Test System initiate the basic call control procedures for an incoming call, to establish a VB connection from the SM-SC to the SM-TE.
	INC_CALL_conf	LTS->LT	The Lower Test System confirms that the requested basic call control procedures for an incoming call have been successfully completed and the VB connection to the SM-TE is established, or that the connection attempt was not successful.
	OUTG_CALL_ind	LTS->LT	Indication that the SM-TE has initiated the basic call control procedures for an outgoing call.
	OUTG_CALL_resp	LT->LTS	Response to the Lower Test System that the basic call control procedures for an outgoing call have to be completed or refused.
	CALL_RELEASE_ind	LTS->LT	Indication from the Lower Test System that the VB connection has been released by the SM-TE.
	CALL_RELEASE_req	LT->LTS	Request the Lower Test System to release the VB connection.
FS	TRANSFER_req	LT->LTS	Request the Lower Test System to transfer an FSK frame to the IUT on the established VB connection.
	TRANSFER_ind	LTS->LT	Indication that the IUT has sent an FSK frame to the tester on the established VB connection.

Tables 3 to 15 contain the descriptions of the ASPs used in the present document, including the ASP parameters (if any) and the kinds of values these may assume. No ASP parameter is optional.

Default values are indicated for most of the ASP parameters. The meaning of "Default value" is:

> If the ASP is applied in a TP behaviour description and no value is indicated explicitly for an ASP parameter, then the application of the default value is assumed. Whenever a value is explicitly indicated for an ASP parameter, this value replaces the default value (at this instance of ASP application).

Table 3: OUTGOING_CALL_req ASP and its parameters

ASP Name: OUTGOING_CALL_req **PCO**: O **Direction**: LT->UT **Description**: Make the SM-TE initiate a VB connection to the SM-SC in order to send an SM (see note 1).		
Parameter	**Default value**	**Description**
SCADDR	TSPX_SC_ADDR	Address of the SM-SC to be called.
SMEID	TSPX_SME_ID	Subaddress of the calling SME.
LENIND	0	Length Indicator **0**: The length of the TL message to be submitted is at most 255 octets long. It is assumed that the SM is not segmented at Data Link layer. **1**: The length of the TL message to be submitted is more than 255 octets long and requires two DLL messages to be sent, i.e. the SM has to be segmented into two segments at Data Link layer. (see note 2) **2**: The length of the TL message to be submitted is more than 2*255 octets long and requires three DLL messages to be sent, i.e. the SM has to be segmented into three segments at Data Link layer. (see note 2)
TFNUM	1	Number of SMs to be submitted during the same VB connection. (see note 2)
Comments: NOTE 1: For the purposes of the DLL tests the type of SM(s) to be submitted (as defined by the Media Identifier) does not matter. NOTE 2: It is assumed that the two mechanisms associated with ASP parameters: LENIND (extension mechanism at DLL) and TFNUM (number of SMs) are independent. This means e.g. that one "long" SM can be transmitted with TFNUM=1 to test the extension mechanism at the DLL (LENIND=1).		

Table 4: SET_MEM_FULL_req ASP and its parameters

ASP Name: SET_MEM_FULL_req **PCO**: O **Direction**: LT->UT **Description**: Request the user to make the SM-TE memory full.		
Parameter	**Default value**	**Description**
Comments:		

Table 5: SET_MEM_FULL_ind ASP and its parameters

ASP Name: SET_MEM_FULL_ind **PCO**: O **Direction**: UT->LT **Description**: Indicates that the SM-TE is in the Memory Full state		
Parameter	**Default value**	**Description**
Comments:		

Table 6: EMPTY_MEM_req ASP and its parameters

ASP Name: EMPTY_MEM_req **PCO**: O **Direction**: LT->UT **Description**: Request the user to delete one or more SMs to make the SM-TE memory again available.		
Parameter	**Default value**	**Description**
Comments:		

Table 7: EMPTY_MEM_ind ASP and its parameters

ASP Name: EMPTY_MEM_ind **PCO**: O **Direction**: UT->LT **Description**: Indicates that the SM-TE is not anymore in the Memory Full state		
Parameter	**Default value**	**Description**
Comments:		

Table 8: INC_CALL_req ASP and its parameters

ASP Name: INC_CALL_req **PCO**: CC **Direction**: LT->LTS **Description**: Make the Lower Test System initiate the basic call control procedures for an incoming call, to establish a VB connection from the SM-SC to the SM-TE.		
Parameter	**Default value**	**Description**
CLDTE	TSPX_CLD_TE	Address of the SM-TE to be called.
SCADDR	TSPX_SC_ADDR	Address of the calling SM-SC.
SMEID	TSPX_SME_ID	Subaddress of the called SME.
Comments:		

Table 9: INC_CALL_conf ASP and its parameters

ASP Name: INC_CALL_conf **PCO**: CC **Direction**: LTS->LT **Description**: The Lower Test System confirms that the requested basic call control procedures for an incoming call have been successfully completed and the VB connection to the SM-TE is established, or that the connection attempt was not successful.		
Parameter	**Default value**	**Description**
ESTCONF	TRUE	**TRUE** if the requested VB connection to the SM-TE/SME has been successfully established. Otherwise **FALSE**.
Comments:		

Table 10: OUTG_CALL_ind ASP and its parameters

ASP Name: OUTG_CALL_ind **PCO**: CC **Direction**: LTS->LT **Description**: Indication that the SM-TE has initiated the basic call control procedures for an outgoing call.		
Parameter	**Default value**	**Description**
SCADDR	TSPX_SC_ADDR	Address of the called SM-SC.
SMEID	TSPX_SME_ID	Subaddress of the calling SME.
Comments:		

Table 11: OUTG_CALL_resp ASP and its parameters

ASP Name: OUTG_CALL_resp **PCO**: CC **Direction**: LT->LTS **Description**: Response to the Lower Test System that the basic call control procedures for an outgoing call have to be completed or refused.		
Parameter	**Default value**	**Description**
ESTRESP	TRUE	TRUE if the outgoing call is to be accepted, otherwise FALSE.
Comments:		

Table 12: CALL_RELEASE_ind ASP and its parameters

ASP Name: CALL_RELEASE_ind **PCO**: CC **Direction**: LTS->LT **Description**: Indication from the Lower Test System that the VB connection has been released by the SM-TE.		
Parameter	**Default value**	**Description**
Comments: No confirmation is issued by the tester. It is a matter of the test system to complete the release procedures. The LTS shall not issue this ASP when the tester has issued a CALL_RELEASE_req before on the same VB connection.		

Table 13: CALL_RELEASE_req ASP and its parameters

ASP Name: CALL_RELEASE_req **PCO**: CC **Direction**: LT->LTS **Description**: Request the Lower Test System to release the VB connection to the SM-TE.		
Parameter	**Default value**	**Description**
Comments: No confirmation is expected by the LT from the LTS. The LTS shall complete the release procedures when receiving this ASP. When this ASP is received by the LTS after VB release has been signalled by the SM-TE (release collision), the LTS shall continue the VB release as if no CALL_RELEASE_req were received.		

Table 14: TRANSFER_req ASP and its parameters

ASP Name: TRANSFER_req **PCO**: FS **Direction**: LT->LTS **Description**: Request the Lower Test System to transfer an FSK frame to the IUT on the established VB connection.		
Parameter	**Default value**	**Description**
CHS	300	Number of alternating Space and Mark bits that shall be transferred by the Lower Test System in front of the DLL PDU, before the Mark signal (Channel Seizure).
MS	80	Number of mark bits that shall be transferred by the Lower Test System in front of the DLL PDU, after the Channel Seizure (Mark signal).
PDU	-	DLL message to be transmitted, as specified in clause 6.5. If the value given to this parameter is OMIT it is assumed that the DLL_SMS_EST (i.e. a null message, composed by only the Channel Seizure and Mark signal) is transmitted.
CS	TRUE	If TRUE, the PDU is transferred as an FSK frame with the correct checksum appended, as defined in clause 6.3.2.1 of ES 201 912 [1]. If FALSE, a checksum error is generated by the Lower Test System (see note)
Comments: NOTE: In case PDU=OMIT, i.e. in case the null message DLL_SMS_EST is transmitted, the checksum does not apply, so that the value of CS is not taken into account.		

Table 15: TRANSFER_ind ASP and its parameters

ASP Name: TRANSFER_ind **PCO**: FS **Direction**: LTS ->LT **Description**: Indication that the IUT has sent a correct FSK frame to the tester on the established VB connection.		
Parameter	**Default value**	**Description**
PDU	-	Received DLL message as specified in clause 6.5. (see note)
Comments: NOTE: A received FSK frame is only passed to the LT by the LTS if the received "Channel Seizure" is composed by 300 bits, the "Mark signal" is in the range of 80 ± 25 mark bits, if the checksum is correct and if the contents of the "Message length" octet is consistent with the actual length of the received message (i.e. the length of the "Payload"). If an FSK frame is received by the LTS which is not correct with respect to one or more of these features, this will lead to a timeout in the test, because the message is not received by the LT. It is a matter of the test system to indicate errors related to these features. If the value given to this parameter is OMIT it is assumed that the DLL_SMS_EST (i.e. a null message, composed by only the Channel Seizure and Mark signal) is received.		

6.4 Test Purpose text formulation

In the textual description, contained in the "Purpose" row of each following table, the notation "DLL_SMS_ACK1" or "DLL_SMS_ACK0" for example, means that these acknowledgement messages do not contain any payload. Otherwise, the notation "DLL_SMS_ACK1 (SMS_SUBMIT_REP)" or "DLL_SMS_ACK0 (SMS_SUBMIT_REP)" for outgoing calls and "DLL_SMS_ACK1 (SMS_DELIVERY_REP)" or "DLL_SMS_ACK0 (SMS_DELIVER_REP)" for incoming calls is used. In brackets the TL message contained in the DLL message is shown. Moreover, the expression "the…message" is used when the number and value of the bytes which constitutes the message are fixed (e.g. the DLL_SMS_ENQ message); while the expression "a…message" is used when they can vary (e.g. a DLL_SMS_INFO (SMS_SUBMIT) message, where the SMS_SUBMIT TL message depends on the SM composed by the user, or a DLL_SMS_ENQ message with wrong checksum, where the wrong checksum byte is different from the correct value).

6.5 UBS2 DLL messages (PDUs)

6.5.1 List of UBS2 DLL messages

Table 16 lists the UBS2 DLL messages used in the present document.

Table 16: List of UBS2 Data Link layer messages

Message name	Purpose description
DLL_SMS_EST	Indicates that the Data Link Layer connection has been established (see note).
DLL_SMS_INFO-MO	Carries the SMS_SUBMIT TL message in case of a SM submission.
DLL_SMS_INFO-STA	Carries the SM-TE_STATUS TL message to communicate to the SM-SC the SM memory availability of the SM-TE.
DLL_SMS_INFO-MT	Carries the SMS_DELIVERY TL message or the SMS_STATUS_REP TL message in case of a SM delivery.
DLL_SMS_ACK1	Gives a positive acknowledgement for a received odd frame. It can carry the SMS_SUBMIT_REP TL message or the SMS_DELIVERY_REP TL message.
DLL_SMS_ACK0	Gives a positive acknowledgement for a received even frame. It can carry the SMS_SUBMIT_REP TL message or the SMS_DELIVERY_REP TL message. It can also carry the SM-TE_CAPABILITY TL message. In this case it can be used by the SM-TE, instead of DLL_SMS_EST, to establish the DLL connection.
DLL_SMS_NACK	Gives a negative acknowledgement, indicating that a Data Link layer error has occurred.
DLL_SMS_ENQ	Used to recover a transmission error or to maintain active the Data Link layer.
DLL_SMS_REL	Carries a Data Link Layer connection release command.
DLL_SMS_UNKNOWN	Unknown message, i.e. a message not being defined in ES 201 912 [1]. This message is only be transmitted by the tester in order to create an error.
NOTE: see table 10, PDU parameter.	

6.5.2 How to interpret message parameters and their values

Following clause 6.3.2 of ES 201 912 [1], table 17 shows the fields are defined for DLL messages.

Table 17: DLL message fields

Octet(s)	Subfield	Short field name	Description
Message Type octet	Extension bit	E	1 bit (most significant bit in Message Type octet). Used for DLL extension mechanism (segmentation and re-assembly of Payloads longer than 255 octets).
	Message type field	MT	7 bits (least significant bits in Message Type octet). Identifies the type of the DLL message.
Message length	-	ML	Length of Payload (in octets).
Payload	-	PL	Octet string of maximum length 255; optional, i.e. not present in all DLL messages.

The short field names in table 17, e.g. **MT**, are the abbreviated field names as they appear in DLL messages occurring in test behaviour descriptions, when appropriate.

NOTE 1: Clause 6.3.2 of ES 201 912 [1] specifies also the "Channel Seizure", the "Mark signal" and the "Checksum". These message elements do not appear in the message structure definition used in the present document, but are treated in the ASPs used for DLL message transmission and reception.

The following principle applies for messages to be **transmitted by the tester**:

- The **Message type field** is not shown explicitly, because it is identified by the message name.
- If the **Extension bit** is not shown explicitly, it is understood that the bit value "0" (no segmentation) is sent.
- The **Message length** is not shown explicitly, the contents of the Message length octet is the binary coding of the number of octets in the payload.
- If the **Payload** is not shown explicitly, it is omitted. If the Payload is shown explicitly, it refers to a Test Parameter (see table 18) to be transmitted.

The following principle applies for messages to be **received from the IUT**:

- The **Message type field** is not shown explicitly, because it is identified by the message name.
- If the **Extension bit** is not shown explicitly, it is understood that the bit value "0" (no segmentation) has to be received.
- The **Message length** is not shown explicitly. The contents of the received Message length octet has to be the binary coding of the number of octets in the payload (otherwise the message is not passed to the LT by the LTS).
- If the **Payload** is not shown explicitly, it is assumed that no Payload has to be received in this message. If the Payload is shown explicitly, it either refers to a Test Parameter (see table 18), or it explicitly indicates "OMIT", or it indicates one of the wildcards **Any** (?) or **AnyOrOmit** (*).

NOTE 2: The wildcards Any and AnyOrOmit and their symbols **?** and ***** have been borrowed from TTCN and have the same semantics:

? means that any non-empty value compatible with the type of the referred field is received,
***** means that either nothing is received, or any non-empty value compatible with the type of the referred field is received.

EXAMPLES:

!DLL_SMS_EST
A correct DLL_SMS_EST message is sent.
!DLL_SMS_INFO-MT (PL=TSPX_SMS_DELIVER_S1)
A syntactically correct DLL_SMS_INFO-MT message is sent, with the Payload represented by Test Parameter "TSPX_SMS_DELIVER_S1", which contains a default TL SMS_DELIVERY message.
?DLL_SMS_INFO-MO (E=1, PL=?)
A syntactically correct DLL_SMS_INFO-MO message is received, where the extension bit is set to "1" (i.e. "further segment follows"), and the Payload is any octet string of length up to 255 octets.

6.6 Behaviour notation

This clause describes the principles used when filling the "**Test description**" entry of the TP tables (see sample in clause 5.7) and the behaviour notation of preambles and postambles.

The notation used to describe the trees containing the required operations and signalling control primitives, is a TTCN-like notation, showing what is sent (character !) and received (character ?) by the tester (playing the role of the SM-SC).

ASPs are sent as shown in the following TTCN-like form:

O!OUTGOING_CALL_req

where **O** denotes the PCO used with the ASP, **!** denotes "transmission" and the **ASP name** follows.

Since default values have been defined for the ASP parameters, parameter values are only indicated when they differ from the default value.

EXAMPLE 1:

O!OUTGOING_CALL_req(TFNUM=2)

In this case 2 SMs are requested to be transmitted by the SM-TE during the VB connection (instead of default 1 SM to be transmitted).

The same principle applies for received ASPs, the symbol **!** being replaced by **?**.

EXAMPLE 2:

CC?OUTG_CALL_ind

In this case the LT receives an indication from the LTS that there is an incoming call from the SM-TE (being requested at the SM-TE as in example 1).

There is one important exception for the presentation of ASPs:

- To increase the readability, ASP names **TRANSFER_req** and **TRANSFER_ind** are not shown, except when a DLL message is to be transmitted with an error in the "Channel Seizure" or the "Mark signal" or the "Checksum".
- DLL message names are directly used instead for transmission and reception. For examples see clause 6.5.2.

Use of timers:

There are two classes of timers assumed in the tester:

a) operational timers, verifying receipt of a response from the IUT or verifying non-reaction of the IUT; and

b) timers used to verify the correct timeout range of protocol timers implemented in the IUT.

Timers of class a) **do not explicitly appear** in the test descriptions of the present document. It is assumed, that the tester starts an operational acknowledge timer when expecting some event from the IUT, and that the timer is stopped when the expected event occurs.

When a protocol timer is claimed to be implemented with a specific timeout value, the tester can never verify the exact timeout value, because of transmission delays and operational delays in the IUT and the tester, but only an accepted time interval. The timers of class b) are therefore normally defined in pairs: one for the lower accepted boundary value and one for the upper boundary value.

As an example, timers TIMER_Tm1_MINUS and TIMER_Tm1_PLUS have been defined for the verification of protocol timer Tm1 implemented in the IUT.

When it has to be verified that Tm1 is in the allowed interval, the following sequence appears in test descriptions (note that in all test descriptions the **expected order** of events is shown):

- START TIMER_Tm1_MINUS
  START TIMER_Tm1_PLUS
  ?TIMEOUT TIMER_Tm1_MINUS
  <expected event>

The meaning of this sequence is:

- At first both timers supervising the boundary values are started. Then the timer for the lower boundary value expires. This verifies that the expected event may not occur too early. Then the expected received event is indicated and no timeout of the timer for the upper boundary value is indicated. This ensures that the expected event occurs before timeout of the timer for the upper boundary value (TIMER_Tm1_PLUS). It is assumed, and not explicitly shown, that TIMER_Tm1_PLUS is stopped when the expected event occurs.

NOTE: In some cases only the lower boundary of a timer has to be verified. E.g. for timer T11min it has only to be verified that the next frame is not transmitted by the IUT **before** timeout of T11min. In this case no pair of test timers is necessary.

To define an allowed timer interval, i.e. lower and upper boundary, for a timeout value claimed to be implemented, Test Parameter TSPX_TDELTA has been defined (see table 18).

Notes are put into the behaviour descriptions whenever it appears to be necessary.

Preambles and Postambles:

The description of Preambles and Postamble starts with an **Objective** definition.

The behaviour description ends with the preamble or postamble name. Between start and end the notation is as described above for "**Test description**".

Each preamble shows the state from where it starts (idle or a different state reached by the execution of another preamble), then it shows the operations executed in this preamble and finally the state or configuration reached, using the notation described above.

Each test purpose description shows the state from where it starts by identifying a preamble.

Notes are put into the behaviour descriptions whenever it appears to be necessary.

6.7 Parameterization and selection

NOTE: During the TSS&TP development Test Parameters have been collected in tables 18 and 19. Test Parameter names starting with "TSPX" are used for test parameterization and will correspond to PIXIT items, TS Parameter names starting with "TSPC" are used for selection and normally correspond to PICS items. Only Test Parameters referred to in the TP description tables appear here. It is assumed that the Test Parameters defined here will be transformed into TTCN "Test Suite Parameters" in an ATS basing on this TSS&TP, presumably in a one-to-one fashion.

Table 18 shows the Test Parameters that are necessary to parameterize the test descriptions to the necessary extent. They will normally appear as ASP parameter values or PDU field contents. It is also possible that they appear in [] brackets as qualifiers for different branches of behaviour (see clause 6.6).

Table 18 specifies the Test Parameter **name**, its **type** (normally a string, integer or Boolean type) and the **explanation** what it is used for.

Table 18: Test Parameters used for parameterization (associated with PIXIT items)

Test Parameter name	Type	Description
TSPX_CLD_TE	OCTETSTRING	Address of the destination SM-TE to be called by the SUT.
TSPX_SC_ADDR1	OCTETSTRING	Address of the SM-SC to be called by the SUT and stored in the SUT.
TSPX_SC_ADDR2	OCTETSTRING	Address of the SM-SC, stored in the SUT, from which the SUT can receive SMs.
TSPX_SME_ID	OCTETSTRING	Subaddress of an SME implemented in the SUT (referred to as SME1). This is the default SME subaddress.
TSPX_SMS_DELIVER_S1	OCTETSTRING	SMS_DELIVERY TL message of length up to 255 octets, that is used as "default" message to be delivered, and is transmitted as Payload in a DLL message.
TSPX_SMS_DELIVER_S2	OCTETSTRING	SMS_DELIVERY TL message of length up to 255 octets, that is used as second message to be delivered on the same VB connection, and is transmitted as Payload in a DLL message.
TSPX_SMS_DELIVER_S3	OCTETSTRING	Segment 1 of length up to 255 octets, of an SMS_DELIVERY TL message that is longer than 255 octets in total. Segment 1 is transmitted as Payload in a DLL message, where the extension bit is set to "1".
TSPX_SMS_DELIVER_S4	OCTETSTRING	Segment 2 (second but not last segment) of length up to 255 octets, of an SMS_DELIVERY TL message that is longer than 255 octets in total. Segment 2 is transmitted as Payload in a DLL message, where the extension bit is set to "1".
TSPX_SMS_DELIVER_S5	OCTETSTRING	Segment 2 (last segment) of length up to 255 octets, of an SMS_DELIVERY TL message that is longer than 255 octets in total and contains a TL error. Segment 2 is transmitted as Payload in a DLL message, where the extension bit is set to "0".
TSPX_SMS_DELIVER_S6	OCTETSTRING	Segment 3 (last segment) of length up to 255 octets, of an SMS_DELIVERY TL message that is longer than 255 octets in total and contains a TL error. Segment 3 is transmitted as Payload in a DLL message, where the extension bit is set to "0".
TSPX_SMS_STATUS_REP	OCTETSTRING	SMS_STATUS_REP TL message of length up to 255 octets, which is transmitted as Payload in a DLL message.
TSPX_SMS_SUBMIT_REP_CONF	OCTETSTRING	SMS_SUBMIT_REP TL message of length up to 255 octets, which can be sent by the tester in a DLL_SMS_ACK1 or DLL_SMS_ACK0 message as a confirmation of a TL message submitted by the SM-TE.

Test Parameter name	Type	Description
TSPX_UNKNOWN_PL	OCTETSTRING	Payload to be transmitted in the DLL_SMS_UNKNOWN message.
TSPX_SMS_SUBMIT	OCTETSTRING	Payload up to 255 octets, to be transmitted in the DLL_SMS_INFO-MO message.
TSPX_TIMEOUT_Tm1	INTEGER	Timeout value implemented for Timer Tm1 (ms). A value between 720 and 880 is allowed (800 ± 10%). See ES 201 912 [1], clause 6.3.2.3, table 6.
TSPX_TIMEOUT_Tm2	INTEGER	Timeout value implemented for Timer Tm1 (ms). A value between 6 840 and 8 360 is allowed (7 600 ± 10%). See ES 201 912 [1], clause 6.3.2.3, table 6.
TSPX_TIMEOUT_Tm3	INTEGER	Timeout value implemented for Timer Tm1 (ms). A value between 6 750 and 8 250 is allowed (7 500±10%). See ES 201 912 [1] clause 6.3.2.3 table 6.
TSPX_TIMEOUT_Tm4	INTEGER	Timeout value implemented for Timer Tm1 (ms). A value between 3 150 and 3 850 is allowed (3 500 ± 10%). See ES 201 912 [1], clause 6.3.2.3, table 6.
TSPX_TIMEOUT_Tm5	INTEGER	Timeout value implemented for Timer Tm1 (ms). A value between 720 and 880 is allowed (800 ± 10%). See ES 201 912 [1], clause 6.3.2.3, table 6.
TSPX_TIMEOUT_Tm6	INTEGER	Timeout value implemented for Timer Tm1 (ms). A value between 180 and 220 is allowed (200 ± 10%). See ES 201 912 [1], clause 6.3.2.3, table 6.
TSPX_AUTOMATIC_INFO_STA_CALL	BOOLEAN	The value is TRUE if the SM-TE, when it leaves the "Memory Full" state following the deletion by the user of one or more SMs stored, automatically establishes an outgoing call to the SM-SC sending the DLL_SMS_INFO_STA message. Otherwise, in case the SM-TE waits for an user indication before establishing such an outgoing call, the value is FALSE.
Comments:		

Table 19 shows the Test Parameters that are necessary to formulate the selection conditions as BOOLEAN expressions. table 19 specifies the Test Parameter **name** (Boolean type), and the **explanation** when it is TRUE or FALSE. These Test Parameters normally correspond to **optional** PICS items.

Table 19: Test Parameters used for selection

Test Parameter name	Description
TSPC_DLL_SM_TE_STATUS_TX	TRUE if the SUT supports the possibility to communicate to the Service Centre the SM memory availability of the SM-TE/SME (ES 202 912-4 [7] (UBS2 PICS) table A.2, Item SMSER10). Otherwise FALSE.
TSPC_DLL_MoreSMs_TX	TRUE if the SUT supports the possibility to send more than one SM within the same VB connection (ES 202 912-4 [7] (UBS2 PICS) table A.2, Item SMSER11). Otherwise FALSE.
TSPC_DLL_MoreSMs_RX	TRUE if the SUT supports the possibility to receive more than one SM within the same VB connection (ES 202 912-4 [7] (UBS2 PICS) table A.2, Item SMSER11). Otherwise FALSE.
Comments:	

Table 20 shows the selection conditions formulated as Boolean expressions. Table 20 specifies the **Selection expression ID** or **name**, the Boolean Expression, and a textual description when a TP carrying this Selection Expression it is applicable for execution with an IUT or not. In its simplest form, the Boolean Expression just refers to a (Boolean) Test Parameter associated with a PICS item. Combinations using Boolean operators like **AND**, **OR** or **NOT** are also allowed.

Table 20: Selection expressions

Selection expression ID	Expression	Description
SEL_DLL_SM-TE_STATUS_TX	TSPC_DLL_SM_TE_STATUS_TX	Selects a TP as applicable if the SUT supports the possibility to communicate to the Service Centre the SM memory availability of the SM-TE/SME.
SEL_DLL_MoreSMs_TX	TSPC_DLL_MoreSMs_TX	Selects a TP as applicable if the SUT supports the possibility to send more than one SM within the same VB connection.
SEL_DLL_MoreSMs_RX	TSPC_DLL_MoreSMs_RX	Selects a TP as applicable if the SUT supports the possibility to receive more than one SM within the same VB connection.
Comments:		

6.8 States

IUT States names are occasionally and informally used as in annex B of ES 201 912 [1].

6.9 Preambles

6.9.1 PRE_INIT

Objective: The preamble has the formal objective to initialize the terminal and test equipment before attempting the next VB connection and DLL connection. A particular behaviour is not associated with this preamble. It is assumed however, that at the end of this preamble no VB connection exists.

6.9.2 PRE_CONN_OUTG

Objective: Establish an outgoing VB connection to the SM-SC. Formal parameters are the Length Indicator requiring "short" or "long" SM(s) to be sent (LengthInd) and the number of SMs to be transmitted during the VB connection (SmsNum).

PRE_CONN_OUTG(LengthInd, SmsNum)
+PRE_INIT
O!OUTGOING_CALL_req(LengthInd, SmsNum)
CC?OUTG_CALL_ind
CC!OUTG_CALL_resp
PRE_CONN_OUTG

6.9.3 PRE_CONN_OUTG_EST

Objective: Establish an outgoing VB connection to the SM-SC, followed by transmission of a DLL_SMS_EST message by the tester (afterwards the IUT is in state CONNECTED). Formal parameters are the Length Indicator requiring "short" or "long" SM(s) to be sent (LengthInd) and the number of SMs to be transmitted during the VB connection (SmsNum).

PRE_CONN_OUTG_EST(LengthInd, SmsNum)
+PRE_CONN_OUTG(LengthInd, SmsNum)
FS!DLL_SMS_EST
PRE_CONN_OUTG_EST

6.9.4 PRE_CONN_OUTG_INFO-MO

Objective: Establish an outgoing VB connection to the SM-SC, followed by transmission of a DLL_SMS_EST message by the tester and a DLL_SMS_INFO-MO message by the IUT (afterwards the IUT is in state RECEIVE). Formal parameters are the Length Indicator requiring "short" or "long" SM(s) to be sent (LengthInd), and the number of SMs to be sent during the same DLL connection (SmsNum). If the LengthInd parameter indicates "long SM", i.e. segmentation necessary, then the IUT sends the DLL_SMS_INFO-MO message with E=1, otherwise with E=0.

PRE_CONN_OUTG_INFO-MO(LengthInd, SmsNum)
+PRE_CONN_OUTG_EST(LengthInd, SmsNum)
[LengthInd=0] FS?DLL_SMS_INFO-MO(E=0, PL=?)
[LengthInd<>0]
FS?DLL_SMS_INFO-MO(E=1, PL=?)
PRE_CONN_OUTG_INFO-MO

6.9.5 PRE_CONN_OUTG_INFO-MO_NO_ANSWER

Objective: Establish an outgoing VB connection to the SM-SC, followed by transmission of a DLL_SMS_EST message by the tester and a DLL_SMS_INFO-MO message by the IUT (afterwards the IUT is in state RECEIVE) in response to which no messages are sent by the tester before the Tm1 timer expiry. Formal parameter is the Length Indicator requiring "short" or "long" SM(s) to be sent (LengthInd). If the LengthInd parameter indicates "long SM", i.e. segmentation necessary, then the IUT sends the DLL_SMS_INFO-MO message with E=1, otherwise with E=0.

PRE_CONN_OUTG_INFO-MO_NO_ANSWER(LengthInd)
+PRE_CONN_OUTG_EST(LengthInd, 1)
[LengthInd=0]
FS?DLL_SMS_INFO-MO(E=0, PL=?)
START TIMER_Tm1_MINUS
START TIMER_Tm1_PLUS
?TIMEOUT TIMER_Tm1_MINUS
[LengthInd<>0]
FS?DLL_SMS_INFO-MO(E=1, PL=?)
START TIMER_Tm1_MINUS
START TIMER_Tm1_PLUS
?TIMEOUT TIMER_Tm1_MINUS
PRE_CONN_OUTG_INFO-MO_NO_ANSWER

6.9.6 PRE_CONN_OUTG_INFO-MO_NACK

Objective: Establish an outgoing VB connection to the SM-SC, followed by transmission of a DLL_SMS_EST message by the tester, a DLL_SMS_INFO-MO message by the IUT and a DLL_SMS_NACK message by the tester (afterwards the IUT is in state SEND). Formal parameter is the Length Indicator requiring "short" or "long" SM(s) to be sent (LengthInd). If the LengthInd parameter indicates "long SM", i.e. segmentation necessary, then the IUT sends the DLL_SMS_INFO-MO message with E=1, otherwise with E=0.

PRE_CONN_OUTG_INFO-MO_NACK(LengthInd
+PRE_CONN_OUTG_EST(LengthInd,1)
[LengthInd=0]
FS?DLL_SMS_INFO-MO(E=0, PL=?)

[LengthInd<>0]
    FS?DLL_SMS_INFO-MO(E=1, PL=?)
FS!DLL_SMS_NACK
PRE_CONN_OUTG_INFO-MO_NACK

6.9.7 PRE_CONN_OUTG_INFO-MO_ACK1

Objective: Establish an outgoing VB connection to the SM-SC, followed by transmission of a DLL_SMS_EST message by the tester, a DLL_SMS_INFO-MO message by the IUT and a DLL_SMS_ACK1 message without payload by the tester (afterwards the IUT is in state SEND). Formal parameter is the Length Indicator requiring "short" or "long" SM(s) to be sent (LengthInd). If the LengthInd parameter indicates "long SM", i.e. segmentation necessary, then the IUT sends the DLL_SMS_INFO-MO message with E=1, otherwise with E=0.

PRE_CONN_OUTG_INFO-MO_ACK1(LengthInd)
+PRE_CONN_OUTG_EST(LengthInd,1)
[LengthInd=0]
    FS?DLL_SMS_INFO-MO(E=0, PL=?)
[LengthInd<>0]
    FS?DLL_SMS_INFO-MO(E=1, PL=?)
FS!DLL_SMS_ACK1
PRE_CONN_OUTG_INFO-MO_ACK1

6.9.8 PRE_CONN_OUTG_INVOKE_INFO-STA

Objective: Request the operator to set the "Memory Full" state in the SM-TE and to delete afterwards one or more SMs stored in the SM-TE, in order to make the SM-TE again ready to accept new incoming SMs. Establish afterwards an outgoing VB connection to the SM-SC (the SM-TE may establish the call automatically or not), followed by transmission of a DLL_SMS_EST message by the tester (afterwards the IUT is in state SEND).

PRE_CONN_OUTG_INVOKE_INFO-STA
O!SET_MEM_FULL_req
O?SET_MEM_FULL_ind
O!EMPTY_MEM_req
[TSPX_AUTOMATIC_INFO_STA_CALL]
    +PRE_INIT
    CC?OUTG_CALL_ind
    CC!OUTG_CALL_resp
    FS!DLL_SMS_EST
[NOT TSPX_AUTOMATIC_INFO_STA_CALL]
O?EMPTY_MEM_ind
    +PRE_CONN_OUTG_EST(0, 1)
PRE_CONN_OUTG_INVOKE_INFO-STA

6.9.9 PRE_CONN_INC

Objective: Establish an incoming VB connection to the SM-TE.

PRE_CONN_INC
+PRE_INIT
CC!INC_CALL_req
CC?INC_CALL_conf
PRE_CONN_INC

6.9.10 PRE_CONN_INC_EST

Objective: Establish an incoming VB connection to the SM-TE, and receive a DLL_SMS_EST message or a DLL_SMS_ACK0(PL=?) sent by the IUT.

PRE_CONN_INC_EST
+PRE_CONN_INC
FS?DLL_SMS_EST
or
FS?DLL_SMS_ACK0(PL=?)
PRE_CONN_INC_EST

6.9.11 PRE_CONN_INC_INFO-MT

Objective: Establish an incoming VB connection to the SM-TE, receive a DLL_SMS_EST message from the IUT and send a DLL_SMS_INFO-MT message by the tester (afterwards the IUT is in state SEND). Formal parameters are the extension bit and the Payload to be sent (Payload).

PRE_CONN_INC_INFO-MT(Ext, Payload)
+PRE_CONN_INC_EST
FS!DLL_SMS_INFO-MT(Ext, PL=Payload)
PRE_CONN_INC_INFO-MT

6.9.12 PRE_CONN_INC_INFO-MT_RESP

Objective: Establish an incoming VB connection to the SM-TE, receive a DLL_SMS_EST message from the IUT, send a DLL_SMS_INFO-MT message by the tester and receive a DLL_SMS_ACK1 message (positive acknowledgement) from the IUT (afterwards the IUT is in state RECEIVE) with or without payload. If the DLL_SMS_ACK1 does not contain a payload, the tester will send a DLL_SMS_ENQ message and receive a DLL_SMS_ACK1 message (positive acknowledgement) from the IUT with or without payload. The tester will stop to send the DLL_SMS_ENQ message after receiving a DLL_SMS_ACK1 message with payload. Formal parameters are the Payload to be sent and the payload to be received (Payload). For this Preamble only a short payload (up to 255 octets) is applicable.

PRE_CONN_INC_INFO-MT_RESP(Payload)
+PRE_CONN_INC_INFO-MT(E=0, Payload)
+RECEIVE_ACK1_WITH_PL
PRE_CONN_INC_INFO-MT_RESP

6.10 Postambles

6.10.1 General

The following postamble is used to ensure that the VB connection is released at the end of a TP. In this postamble the tester is the initiator (of the release).

In case the IUT does not release when it is expected to do so, it is assumed that the tester will ultimately initiate the release. However this is not shown.

In some cases the release can already occur in the TP body. In this case the use of the postamble is not intended to repeat release signalling.

6.10.2 POST_TESTER_RELEASE_VB

Objective: Release of the VB connection by the tester, independently of what the status of the DLL connection is and who is the originator of the call.

POST_TESTER_RELEASE_VB
CC!CALL_RELEASE_req
POST_TESTER_RELEASE_VB

6.11 Other test steps

6.11.1 General

The test steps of this clause have been defined to comprise sequences of more than one event, occurring several times and being referred to as one test step in preambles or test description bodies.

6.11.2 RECEIVE_ACK1_WITH_PL

Objective: Verify receipt of a DLL_SMS_ACK1 message with payload from the IUT. Up to 49 DLL_SMS_ACK1 messages without payload may be received before that.

RECEIVE_ACK1_WITH_PL
START TIMER_Tm6_MINUS
START TIMER_Tm6_PLUS
Either:

    FS?DLL_SMS_ACK1(PL=?)
    Or
    ?TIMEOUT TIMER_Tm6_MINUS
        FS?DLL_SMS_ACK1(PL=?)
        Or
        FS?DLL_SMS_ACK1
End either
Set COUNTER=0
While (COUNTER < 50) AND (Payload received in DLL_SMS_ACK1 =OMIT)
Set COUNTER=COUNTER+1
START TIMER_Tm5_MINUS
?TIMEOUT TIMER_Tm5_MINUS
FS!DLL_SMS_ENQ
FS?DLL_SMS_ACK1(PL=*)
End While

    NOTE: If COUNTER has the value of 50 and no DLL_SMS_ACK1 message with payload has been received, then the outcome of this step is unsuccessful, which gives a FAIL verdict to the test applying this step.

RECEIVE_ACK1_WITH_PL

6.11.3 RECEIVE_ACK0_WITH_PL

Objective: Verify receipt of a DLL_SMS_ACK0 message with payload from the IUT. Up to 49 DLL_SMS_ACK0 messages without payload may be received before that.

RECEIVE_ACK0_WITH_PL
START TIMER_Tm6_MINUS
START TIMER_Tm6_PLUS
Either:
    FS?DLL_SMS_ACK0(PL=?)
    Or
    ?TIMEOUT TIMER_Tm6_MINUS
        FS?DLL_SMS_ACK0(PL=?)

Or
FS?DLL_SMS_ACK0
End either
Set COUNTER=0
While (COUNTER < 50) AND (Payload received in DLL_SMS_ACK0 =OMIT)
Set COUNTER=COUNTER+1
START TIMER_Tm5_MINUS
?TIMEOUT TIMER_Tm5_MINUS
FS!DLL_SMS_ENQ
FS?DLL_SMS_ACK0(PL=*)
End While

NOTE: If COUNTER has the value of 50 and no DLL_SMS_ACK0 message with payload has been received, then the outcome of this step is unsuccessful, which gives a FAIL verdict to the test applying this step.

RECEIVE_ACK0_WITH_PL

7 Test purpose descriptions

NOTE: Requirements references are for ES 201 912 [1].

7.1 Test purposes for Frame transfer/Synchronization (UBS2_DLL_FRM_SYNC)

	UBS2_DLL_FRM_SYNC_VAL_01
Purpose:	Verify that the SM-TE correctly receives a DLL_SMS_EST message with a Channel Seizure length of 300 alternating Space and Mark bits
Requirements refs:	6.3.2.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=OMIT) FS?DLL_SMS_INFO-MO(PL=?)
Pass criteria:	DLL_SMS_INFO-MO received, indicating correct receipt of DLL_SMS_EST by the IUT
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_SYNC_VAL_02
Purpose:	Verify that the SM-TE correctly receives a DLL_SMS_INFO-MT message with a Channel Seizure length of 300 alternating Space and Mark bits
Requirements refs:	6.3.2.1
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_INFO-MT(TSPX_SMS_DELIVER_S1), CS=TRUE) FS?DLL_SMS_ACK1(PL=*)
Pass criteria:	DLL_SMS_ACK1(PL=*) received, indicating correct receipt of DLL_SMS_INFO-MT by the IUT
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_SYNC_VAL_03
Purpose:	Verify that the SM-TE correctly receives a DLL_SMS_EST message with a Mark Signal length of 55 Mark bits
Requirements refs:	6.3.2.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(CHS=300, MS=55, PDU=OMIT) FS?DLL_SMS_INFO-MO(E=0, PL=?) Or FS?DLL_SMS_INFO-STA(PL=?)
Pass criteria:	DLL_SMS_INFO-MO or DLL_SMS_INFO_STA received, indicating correct receipt of DLL_SMS_EST
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_SYNC_VAL_04
Purpose:	Verify that the SM-TE correctly receives a DLL_SMS_INFO-MT message with a Mark Signal length of 55 Mark bits
Requirements refs:	6.3.2.1
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!TRANSFER_req(CHS=300, MS=55, PDU=DLL_SMS_INFO-MT(TSPX_SMS_DELIVER_S1), CS=TRUE) FS?DLL_SMS_ACK1(PL=*)
Pass criteria:	DLL_SMS_ACK1(PL=*) received, indicating correct receipt of DLL_SMS_INFO-MT by the IUT
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_SYNC_VAL_05
Purpose:	Verify that the SM-TE correctly receives a DLL_SMS_EST message with a Mark Signal length of 105 Mark bits
Requirements refs:	6.3.2.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(CHS=300, MS=105, PDU=OMIT) FS?DLL_SMS_INFO-MO(E=0, PL=?) Or FS?DLL_SMS_INFO-STA(PL=?)
Pass criteria:	DLL_SMS_INFO-MO or DLL_SMS_INFO_STA received, indicating correct receipt of DLL_SMS_EST
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_SYNC_VAL_06
Purpose:	Verify that the SM-TE correctly receives a DLL_SMS_INFO-MT message with a Mark Signal length of 105 Mark bits
Requirements refs:	6.3.2.1
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!TRANSFER_req(CHS=300, MS=105, PDU=DLL_SMS_INFO-MT(TSPX_SMS_DELIVER_S1), CS=TRUE) FS?DLL_SMS_ACK1(PL=*)
Pass criteria:	DLL_SMS_ACK1(PL=*) received, indicating correct receipt of DLL_SMS_INFO-MT by the IUT
Postamble:	POST_TESTER_RELEASE_VB

7.2 Test purposes for Frame transfer/Timing intervals

	UBS2_DLL_FRM_TIM_VAL_01
Purpose:	Verify that the SM-TE sends the DLL_SMS_EST message or the DLL_SMS_ACK0 (SM-TE_CAPABILITY) message not before T10min expiry and not after T3 expiry, the timers being started after the SM call establishment.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_INC
Test description:	START TIMER_T10min_MINUS START TIMER_T3_PLUS ?TIMEOUT TIMER_T10min_MINUS FS?TRANSFER_ind(PDU=OMIT) Or FS?DLL_SMS_ACK0(PL=?)
Pass criteria:	DLL_SMS_EST or DLL_SMS_ACK0 with payload received after timeout TIMER_T10min_MINUS and before timeout TIMER_T3_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_02
Purpose:	Verify that the SM-TE sends a DLL_SMS_INFO-MO (SMS_SUBMIT) message not before T11min expiry and not after T2 expiry, the timers being started after the reception of the DLL_SMS_EST message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_EST(LengthInd=0, SmsNum=1)
Test description:	START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_INFO-MO(E=0, PL=?)
Pass criteria:	DLL_SMS_INFO-MO received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_03
Purpose:	Verify that the SM-TE sends a DLL_SMS_INFO-STA (SM-TE_STATUS) message not before T11min expiry and not after T2 expiry, the timers being started after the reception of the DLL_SMS_EST message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	SEL_DLL_SM-TE_STATUS_TX
Preamble:	PRE_CONN_OUTG_INVOKE_INFO-STA
Test description:	START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_INFO-STA(PL=?)
Pass criteria:	DLL_SMS_INFO-STA received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_04
Purpose:	Verify that in case of a TL message larger than 255 octets (two segments), the SM-TE, having received the DLL_SMS_ACK1 message in response to the first DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1", sends the second DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="0" not before T11min expiry and not after T2 expiry, the timers being started after the reception of the DLL_SMS_ACK1 message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=1, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1 START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_INFO-MO(E=0, PL=?)
Pass criteria:	DLL_SMS_INFO-MO(E=0) received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_05
Purpose:	Verify that in case of a TL message larger than 255 octets (three segments), the SM-TE, having received the DLL_SMS_ACK1 message in response to the first DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1" and the DLL_SMS_ACK0 message in response to the second DLL_SMS_INFO-MO with extension bit="1", sends the third DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="0" not before T11min expiry and not after T2 expiry, the timers being started after the reception of the DLL_SMS_ACK0 message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO_ACK1(LengthInd=2)
Test description:	FS?DLL_SMS_INFO-MO(E=1, PL=?) FS!DLL_SMS_ACK0 START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_INFO-MO(E=0, PL=?)
Pass criteria:	DLL_SMS_INFO-MO(E=0) received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_06
Purpose:	Verify that the SM-TE either sends the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message, not before T11min expiry and not after Tm6 expiry, or sends the DLL_SMS_ACK1 message at Tm6 expiry, the timers being started after the reception of a DLL_SMS_INFO-MT (SMS_DELIVERY) message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 6 and 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=0, TSPX_SMS_DELIVER_S1)
Test description:	START TIMER_T11min_MINUS START TIMER_Tm6_MINUS START TIMER_Tm6_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ACK1(PL=?) Or ?TIMEOUT TIMER_Tm6_MINUS FS?DLL_SMS_ACK1(PL=?) Or FS?DLL_SMS_ACK1
Pass criteria:	DLL_SMS_INFO-ACK1(PL=?) received after TIMER_T11min_MINUS and before timeout TIMER_Tm6_MINUS or DLL_SMS_INFO-ACK1(PL=*) received before timeout TIMER_Tm6_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_07
Purpose:	Verify that the SM-TE sends the DLL_SMS_ ACK1 message not before T11min expiry and not after T1 expiry, the timers being started after the reception of a first DLL_SMS_INFO-MT (SMS_DELIVERY) message with extension bit="1".
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=1, TSPX_SMS_DELIVER_S3)
Test description:	START TIMER_T11min_MINUS START TIMER_T1_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ACK1
Pass criteria:	DLL_SMS_ACK1 received after timeout TIMER_T11min_MINUS and before timeout TIMER_T1_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_08
Purpose:	Verify that the SM-TE, having acknowledged a first DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message with the DLL_SMS_ACK1 message, sends the DLL_SMS_ ACK0 message not before T11min expiry and not after T1 expiry, the timers being started after the reception of a second DLL_SMS_INFO-MT (SMS_DELIVERY) message with extension bit="1".
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=1, TSPX_SMS_DELIVER_S3)
Test description:	FS?DLL_SMS_ACK1 FS!DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S4) START TIMER_T11min_MINUS START TIMER_T1_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 received after timeout TIMER_T11min_MINUS and before timeout TIMER_T1_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_09
Purpose:	Verify that the SM-TE, having acknowledged a first DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message with the DLL_SMS_ACK1 message and a second DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message with the DLL_SMS_ACK0 message, either sends the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message, not before T11min expiry and not after Tm6 expiry, or sends the DLL_SMS_ACK1 message at Tm6 expiry, the timers being started after the reception of a third DLL_SMS_INFO-MT (SMS_DELIVERY) message with extension bit="0".
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=1, TSPX_SMS_DELIVER_S3)
Test description:	FS?DLL_SMS_ACK1 FS!DLL_SMS_INFO-MT(E=1, PL=TSPX_SMS_DELIVER_S4) FS?DLL_SMS_ACK0 FS!DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S6) START TIMER_T11min_MINUS START TIMER_Tm6_MINUS START TIMER_Tm6_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ACK1(PL=?) Or ?TIMEOUT TIMER_Tm6_MINUS FS?DLL_SMS_ACK1(PL=?) Or FS?DLL_SMS_ACK1
Pass criteria:	DLL_SMS_INFO-ACK1(PL=?) received after timeout TIMER_T11min_MINUS and before timeout TIMER_Tm6_MINUS or DLL_SMS_INFO-ACK1(PL=*) received before timeout TIMER_Tm6_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_10
Purpose:	Verify that the SM-TE either sends the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message, not before T11min expiry and not after Tm6 expiry, or sends the DLL_SMS_ACK1 message at Tm6 expiry, the timers being started after the reception of a DLL_SMS_INFO-MT (SMS_STATUS_REP) message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 6 and 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=0, TSPX_SMS_STATUS_REP)
Test description:	START TIMER_T11min_MINUS START TIMER_Tm6_MINUS START TIMER_Tm6_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ACK1(PL=?) Or ?TIMEOUT TIMER_Tm6_MINUS FS?DLL_SMS_ACK1(PL=?) Or FS?DLL_SMS_ACK1
Pass criteria:	DLL_SMS_INFO-ACK1(PL=?) received after timeout TIMER_T11min_MINUS and before timeout TIMER_Tm6_MINUS or DLL_SMS_INFO-ACK1(PL=*) received before timeout TIMER_Tm6_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_11
Purpose:	Verify that the SM-TE sends the DLL_SMS_NACK message not before T11min expiry and not after T1 expiry, the timers being started after the reception of a DLL_SMS_INFO-MT (SMS_DELIVERY) message with wrong checksum.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_INFO-MT(TSPX_SMS_DELIVER_S1), CS=FALSE) START TIMER_T11min_MINUS START TIMER_T1_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_NACK
Pass criteria:	DLL_SMS_NACK received after timeout TIMER_T11min_MINUS and before timeout TIMER_T1_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_12
Purpose:	Verify that the SM-TE sends the DLL_SMS_NACK message not before T11min expiry and not after T1 expiry, the timers being started after the reception of a DLL_SMS_INFO-MT (SMS_STATUS_REP) message with wrong checksum.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_INFO-MT(TSPX_SMS_STATUS_REP), CS=FALSE) START TIMER_T11min_MINUS START TIMER_T1_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_NACK
Pass criteria:	DLL_SMS_NACK received after timeout TIMER_T11min_MINUS and before timeout TIMER_T1_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_13
Purpose:	Verify that the SM-TE sends the DLL_SMS_NACK message not before T11min expiry and not after T1 expiry, the timers being started after the reception of a DLL_SMS_ENQ message with wrong checksum.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ENQ, CS=FALSE) START TIMER_T11min_MINUS START TIMER_T1_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_NACK
Pass criteria:	DLL_SMS_NACK received after timeout TIMER_T11min_MINUS and before timeout TIMER_T1_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_14
Purpose:	Verify that the SM-TE sends the DLL_SMS_NACK message not before T11min expiry and not after T1 expiry, the timers being started after the reception of a DLL_SMS_REL message with wrong checksum.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_REL, CS=FALSE) START TIMER_T11min_MINUS START TIMER_T1_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_NACK
Pass criteria:	DLL_SMS_NACK received after timeout TIMER_T11min_MINUS and before timeout TIMER_T1_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_15
Purpose:	Verify that the SM-TE sends again the last DLL_SMS_INFO-MO (SMS_SUBMIT) message sent, not before T11min expiry and not after T2 expiry, the timers being started after the reception of a DLL_SMS_NACK message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO_NACK(LengthInd=0)
Test description:	START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_INFO-MO(E=0, PL=?)
Pass criteria:	DLL_SMS_INFO-MO(E=0, PL=?) received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_16
Purpose:	Verify that the SM-TE sends again the last DLL_SMS_INFO-STA (SM-TE_STATUS) message sent, not before T11min expiry and not after T2 expiry, the timers being started after the reception of a DLL_SMS_NACK message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INVOKE_INFO-STA
Test description:	FS!DLL_SMS_NACK START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_INFO-STA(PL=?)
Pass criteria:	DLL_SMS_INFO-MO(E=0, PL=?) received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_17
Purpose:	Verify that the SM-TE sends again the last DLL_SMS_ENQ message sent, not before T11min expiry and not after T2 expiry, the timers being started after the reception of a DLL_SMS_NACK message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO_NO_ANSWER(LengthInd=0)
Test description:	FS?DLL_SMS_ENQ FS!DLL_SMS_NACK START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_18
Purpose:	Verify that the SM-TE sends again the last DLL_SMS_REL message sent, not before T11min expiry and not after T2 expiry, the timers being started after the reception of a DLL_SMS_NACK message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!DLL_SMS_NACK START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_REL
Pass criteria:	The second DLL_SMS_REL received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_19
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, sends the DLL_SMS_ENQ message not before T11min expiry and not after T2 expiry, the timers being started after the reception of a DLL_SMS_ACK1 message with wrong checksum.
Requirements refs:	6.3.1, 6.3.2.1, table 5 and, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK1, CS=FALSE) START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_20
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message acknowledged with the DLL_SMS_ACK1 message and having sent after the DLL_SMS_ENQ message, sends again the DLL_SMS_ENQ message not before T11min expiry and not after T2 expiry, the timers being started after the reception of a DLL_SMS_ACK1 message with wrong checksum.
Requirements refs:	6.3.1, 6.3.2.1, table 5 and, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO_ACK1(LengthInd=0)
Test description:	FS?DLL_SMS_ENQ FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK1, CS=FALSE) START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_21
Purpose:	Verify that the SM-TE sends the DLL_SMS_REL message not before T11min expiry and not after T2 expiry, the timers being started after the reception of a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message in response to a DLL_SMS_INFO-MO (SMS_SUBMIT) message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_22
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message acknowledged with the DLL_SMS_ACK1 (SMS_SUBMIT_REP) message and having sent the DLL_SMS_REL message afterwards, sends again the DLL_SMS_REL message, not before T11min expiry and not after T2 expiry, the timers being started after the reception of a DLL_SMS_ACK0 message with wrong checksum.
Requirements refs:	6.3.1, 6.3.2.1, table 5 and, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK0, CS=FALSE) START TIMER_T11min_MINUS START TIMER_T2_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_REL
Pass criteria:	The second DLL_SMS_REL received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_23
Purpose:	Verify that the SM-TE, having acknowledged the previous DLL_SMS_INFO-MT (SMS_DELIVERY) message with the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message, sends again the DLL_SMS_ACK1 (SMS_DELIVERY) message previously sent, not before T11min expiry and not after T1 expiry, the timers being started after the reception of the DLL_SMS_ENQ message, or, having acknowledged the previous DLL_SMS_INFO-MT (SMS_DELIVERY) message with the DLL_SMS_ACK1 message, sends again the DLL_SMS_ACK1 message or the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message, not before T11min expiry and not after T1 expiry, the timers being started after the reception of the DLL_SMS_ENQ message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=0, TSPX_SMS_DELIVER_S1)
Test description:	FS?DLL_SMS_ACK1(PL=*) If PL=? FS!DLL_SMS_ENQ START TIMER_T11min_MINUS START TIMER_T1_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ACK1(PL=?) Else If PL=OMIT FS!DLL_SMS_ENQ START TIMER_T11min_MINUS START TIMER_T1_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ACK1(PL=*) End if
Pass criteria:	DLL_SMS_ACK1(PL=?) or DLL_SMS_ACK1 received after timeout TIMER_T11min_MINUS and before timeout TIMER_T2_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_24
Purpose:	Verify that the SM-TE, having acknowledged a first DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message with the DLL_SMS_ACK1 message and a second DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message with the DLL_SMS_ACK0 message, sends again the DLL_SMS_ACK0 message previously sent, not before T11min expiry and not after T1 expiry, the timers being started after the reception of the DLL_SMS_ENQ message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=1, TSPX_SMS_DELIVER_S3)
Test description:	FS?DLL_SMS_ACK1 FS!DLL_SMS_INFO-MT(E=1, ML=Len(TSPX_SMS_DELIVER_S4), PL=TSPX_SMS_DELIVER_S4) FS?DLL_SMS_ACK0 FS!DLL_SMS_ENQ START TIMER_T11min_MINUS START TIMER_T1_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 received after timeout TIMER_T11min_MINUS and before timeout TIMER_T1_PLUS
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_FRM_TIM_VAL_25
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message in response to a DLL_SMS_INFO-MT (SMS_DELIVERY) message or to a following DLL_SMS_ENQ message, sends the DLL_SMS_ACK0 message, not before T11min expiry and not after T1 expiry, the timers being started after the reception of the DLL_SMS_REL message.
Requirements refs:	6.3.1, 6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!DLL_SMS_REL START TIMER_T11min_MINUS START TIMER_T1_PLUS ?TIMEOUT TIMER_T11min_MINUS FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 received after timeout TIMER_T11min_MINUS and before timeout TIMER_T1_PLUS
Postamble:	POST_TESTER_RELEASE_VB

7.3 Test purposes for Outgoing call/Establishment (UBS2_DLL_OUT_EST)

	UBS2_DLL_OUT_EST_VAL_01
Purpose:	Verify that the SM-TE accepts the DLL_SMS_EST message as first DLL message after that the VB connection has been established (i.e. it stops the Tm3 timer and sends the first frame).
Requirements refs:	6.3.2.1, table 5, B.1.2.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_EST FS?DLL_SMS_INFO-MO(E=0, PL=?)
Pass criteria:	DLL_SMS_INFO-MO message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_EST_VAL_02
Purpose:	Verify that the SM-TE, in case it does not receive any DLL message before the expiry of Tm3 starting from the end of the dialling phase, hangs on.
Requirements refs:	6.3.2.1, table 5, B.1.2.3
Selection Cond.:	
Preamble:	PRE_INIT
Test description:	O!OUTGOING_CALL_req(0,1) CC?OUTG_CALL_ind START TIMER_Tm3_MINUS START TIMER_Tm3_PLUS CC!OUTG_CALL_resp ?TIMEOUT TIMER_Tm3_MINUS CC?CALL_RELEASE_ind
Pass criteria:	VB release indication received after timeout TIMER_Tm3_MINUS and before timeout TIMER_Tm3_PLUS.
Postamble:	None

7.4 Test purposes for Outgoing call/Data transfer (UBS2_DLL_OUT_DAT)

	UBS2_DLL_OUT_DAT_VAL_01
Purpose:	Verify that in case of an SM call established to submit a TL message not larger than 255 bytes, the SM-TE sends a DLL_SMS_INFO-MO (SMS_SUBMIT) with extension bit="0" message after having received the DLL_SMS_EST message.
Requirements refs:	6.3.2.1, table 5, B.1.3.1 and B.1.3.4
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_EST(LengthInd=0, SmsNum=1)
Test description:	FS?DLL_SMS_INFO-MO(E=0, PL=?)
Pass criteria:	DLL_SMS_INFO-MO message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_02
Purpose:	Verify that the SM-TE sends the DLL_SMS_INFO-STA (SM-TE_STATUS) message after having received the DLL_SMS_EST message.
Requirements refs:	6.3.2.1, table 5, B.1.3.1 and B.1.3.4
Selection Cond.:	SEL_DLL_SM-TE_STATUS_TX
Preamble:	PRE_CONN_OUTG_INVOKE_INFO-STA
Test description:	FS?DLL_SMS_INFO-STA(PL=?)
Pass criteria:	DLL_SMS_INFO-STA message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_03
Purpose:	Verify that in case of a TL message larger than 255 octets (two segments), the SM-TE, having received the DLL_SMS_EST message, sends the first DLL_SMS_INFO-MO (SMS_SUBMIT) message with the extension bit="1" and after the reception of the DLL_SMS_ACK1 message in response, sends the second DLL_SMS_INFO (SMS_SUBMIT) message with the extension bit="0".
Requirements refs:	6.3.2.1, §4 and, table 5
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_EST(LengthInd=1, SmsNum=1)
Test description:	FS?DLL_SMS_INFO-MO(E=1, PL=?) FS!DLL_SMS_ACK1 FS?DLL_SMS_INFO-MO(E=0, PL=?)
Pass criteria:	Two DLL_SMS_INFO-MO messages received, the first one with extension bit "1" and the second with extension bit "0".
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_04
Purpose:	Verify that in case of a TL message larger than 255 octets (three segments), the SM-TE, having received the DLL_SMS_ACK1 message in response to the first DLL_SMS_INFO-MO (SMS_SUBMIT) message with the extension bit="1", sends the second DLL_SMS_INFO (SMS_SUBMIT) message with the extension bit="1" and, after the reception of the DLL_SMS_ACK0 message in response, sends the third DLL_SMS_INFO (SMS_SUBMIT) message with the extension bit="0".
Requirements refs:	6.3.2.1, §4 and, table 5
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_EST(LengthInd=2, SmsNum=1)
Test description:	FS?DLL_SMS_INFO-MO(E=1, PL=?) FS!DLL_SMS_ACK1 FS?DLL_SMS_INFO-MO(E=1, PL=?) FS!DLL_SMS_ACK0 FS?DLL_SMS_INFO-MO(E=0, PL=?)
Pass criteria:	DLL_SMS_INFO-MO (E=1, PL) received as the first and second INFO messages and DLL_SMS_INFO-MO (E=0, PL) received as the third INFO message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_05
Purpose:	Verify that, in the case of two SMs to send, the SM-TE, having received a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message in response to the first DLL_SMS_INFO-MO (SMS_SUBMIT) message, sends the second DLL_SMS_INFO-MO (SMS_SUBMIT) message.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	SEL_DLL_MoreSMs_TX
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=2)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_INFO-MO(E=0, PL=?)
Pass criteria:	DLL_SMS_INFO-MO (E=0, PL=?) received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_06
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, without receiving any DLL message in response, sends the DLL_SMS_ENQ message at Tm1 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, B.1.3.7 and B.1.3.8
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ received after timeout of TIMER_Tm1_MINUS and before timeout of TIMER_Tm1_PLUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_07
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-STA (SM-TE_STATUS) message, without receiving any DLL message in response, sends the DLL_SMS_ENQ message at Tm1 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, B.1.3.7 and B.1.3.8
Selection Cond.:	SEL_DLL_SM-TE_STATUS_TX
Preamble:	PRE_CONN_OUTG_INVOKE_INFO-STA
Test description:	FS?DLL_SMS_INFO-STA(PL=?) START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ message received after timeout of TIMER_Tm1_MINUS and before timeout of TIMER_Tm1_PLUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_08
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, without receiving any response, and then the DLL_SMS_ENQ message at Tm1 expiry, without receiving any response, sends again the DLL_SMS_ENQ message at Tm1 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, B.1.3.10 and B.1.3.11
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_ENQ START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_ENQ
Pass criteria:	The second DLL_SMS_ENQ message received after timeout of TIMER_Tm1_MINUS and before timeout of TIMER_Tm1_PLUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_09
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, without receiving any response, and having sent afterwards a first DLL_SMS_ENQ message at Tm1 expiry, without receiving any response, and then a second DLL_SMS_ENQ message at Tm1 expiry, without receiving any response, sends again the DLL_SMS_ENQ message at Tm1 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, 6.3.2.4, table 8, B.1.3.10 and B.1.3.11
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_ENQ START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_ENQ START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_ENQ
Pass criteria:	The third DLL_SMS_ENQ messages received after timeout of TIMER_Tm1_MINUS and before timeout of TIMER_Tm1_PLUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_10
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, without receiving any response, and having sent afterwards a first DLL_SMS_ENQ message at Tm1 expiry, without receiving any response, then a second DLL_SMS_ENQ message at Tm1 expiry, without receiving any response, and finally a third DLL_SMS_ENQ message at Tm1 expiry, without receiving any response, hangs on at Tm1 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.4, table 8, B.1.3.10 and B.1.3.11
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_ENQ START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_ENQ START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_ENQ START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS CC?CALL_RELEASE_ind
Pass criteria:	VB release indication received after timeout of Tm1_MINUS and before timeout of Tm1_PLUS, starting from the third DLL_SMS_ENQ message.
Postamble:	None

	UBS2_DLL_OUT_DAT_VAL_11
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_INFO-MO (SMS_SUBMIT) message and received the DLL_SMS_NACK message in response, sends again the DLL_SMS_INFO-MO (SMS_SUBMIT) message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.3.5
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_NACK FS?DLL_SMS_INFO-MO(E=0, PL=?) (see note)
Pass criteria:	DLL_SMS_INFO-MO message received with extension bit "0" and the same payload as received with the DLL_SMS_INFO-MO message in the preamble.
Postamble:	POST_TESTER_RELEASE_VB
NOTE: The payload received must be the same as the payload received with the DLL_SMS_INFO-MO message in the preamble.	

	UBS2_DLL_OUT_DAT_VAL_12
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_INFO-STA (SM-TE_STATUS) message and received the DLL_SMS_NACK message in response, sends again the DLL_SMS_INFO-STA (SM-TE_STATUS) message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.3.5
Selection Cond.:	SEL_DLL_SM-TE_STATUS_TX
Preamble:	PRE_CONN_OUTG_INVOKE_INFO-STA
Test description:	FS?DLL_SMS_INFO-STA(PL=?) FS!DLL_SMS_NACK FS?DLL_SMS_INFO-STA(PL=?) (see note)
Pass criteria:	The second DLL_SMS_INFO-STA message received with the same payload as the first DLL_SMS_INFO-STA message.
Postamble:	POST_TESTER_RELEASE_VB
NOTE: The payload received must be the same as the payload received with the first DLL_SMS_INFO-STA message.	

	UBS2_DLL_OUT_DAT_VAL_13
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message and received the DLL_SMS_NACK message in response, and having sent again the same DLL_SMS_INFO-MO (SMS_SUBMIT) message afterwards and received the DLL_SMS_NACK message in response, sends again the DLL_SMS_INFO-MO (SMS_SUBMIT) message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.3.9
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_NACK FS?DLL_SMS_INFO-MO(E=0, PL=?) (see note 1) FS!DLL_SMS_NACK FS?DLL_SMS_INFO-MO(E=0, PL=?) (see note 2)
Pass criteria:	DLL_SMS_INFO-MO received two times after DLL_SMS_NACK, with the same payload as received in the DLL_SMS_INFO-MO message in the preamble.
Postamble:	POST_TESTER_RELEASE_VB
NOTE 1: The payload received must be the same as the payload received with the DLL_SMS_INFO-MO message in the preamble. NOTE 2: The payload received must be the same as the payload received with the DLL_SMS_INFO-MO message in the preamble and with the previous message.	

	UBS2_DLL_OUT_DAT_VAL_14
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message and received the DLL_SMS_NACK message in response, and having sent again the same DLL_SMS_INFO-MO (SMS_SUBMIT) message afterwards and received the DLL_SMS_NACK message in response, and finally for the third time the DLL_SMS_INFO-MO (SMS_SUBMIT) message and received the DLL_SMS_NACK message in response, hangs on.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.3.9
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_NACK FS?DLL_SMS_INFO-MO(E=0, PL=?) (see note 1) Note 1: The payload received must be the same as the payload received with the DLL_SMS_INFO-MO message in the preamble. FS!DLL_SMS_NACK FS?DLL_SMS_INFO-MO(E=0, PL=?) (see note 2) FS!DLL_SMS_NACK CC?CALL_RELEASE_ind
Pass criteria:	CALL_RELEASE_ind received after having transmitted DLL_SMS_NACK for the third time.
Postamble:	POST_TESTER_RELEASE_VB
NOTE 1: The payload received must be the same as the payload received with the DLL_SMS_INFO-MO message in the preamble. NOTE 2: The payload received must be the same as the payload received with the DLL_SMS_INFO-MO message in the preamble and with the previous message.	

	UBS2_DLL_OUT_DAT_VAL_15
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, without receiving any response, and having sent afterwards the DLL_SMS_ENQ message at Tm1 expiry and received the DLL_SMS_NACK message in response, sends again the DLL_SMS_ENQ message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, tables 6 and 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	START TIMER_Tm1_PLUS FS?DLL_SMS_ENQ FS!DLL_SMS_NACK FS?DLL_SMS_ENQ
Pass criteria:	The second DLL_SMS_ENQ message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_16
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, without receiving any response, and having sent afterwards a first DLL_SMS_ENQ message at Tm1 expiry, having received the DLL_SMS_NACK message in response, then a second DLL_SMS_ENQ message, having received the DLL_SMS_NACK message in response, sends again the DLL_SMS_ENQ message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, 6.3.2.4, table 8
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	START TIMER_Tm1_PLUS FS?DLL_SMS_ENQ FS!DLL_SMS_NACK FS?DLL_SMS_ENQ FS!DLL_SMS_NACK FS?DLL_SMS_ENQ
Pass criteria:	The third DLL_SMS_ENQ message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_17
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, without receiving any response, and having sent afterwards a first DLL_SMS_ENQ message at Tm1 expiry, having received the DLL_SMS_NACK message in response, then a second DLL_SMS_ENQ message, having received the DLL_SMS_NACK message in response, and finally a third DLL_SMS_ENQ message, having received the DLL_SMS_NACK message in response, hangs on.
Requirements refs:	6.3.2.1, table 5, 6.3.2.4, table 8
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	START TIMER_Tm1_PLUS FS?DLL_SMS_ENQ FS!DLL_SMS_NACK FS?DLL_SMS_ENQ FS!DLL_SMS_NACK FS?DLL_SMS_ENQ FS!DLL_SMS_NACK CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after 3rd DLL_SMS_NACK.
Postamble:	None

	UBS2_DLL_OUT_DAT_VAL_18
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message and received the DLL_SMS_ACK1 message in response, sends the DLL_SMS_ENQ message at Tm5 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, B.1.3.2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1 START TIMER_Tm5_MINUS START TIMER_Tm5_PLUS ?TIMEOUT TIMER_Tm5_MINUS FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ received after timeout of TIMER_Tm5_MINUS and before timeout of TIMER_Tm5_PLUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_19
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-STA (SM-TE_STATUS) message and received the DLL_SMS_ACK1 message in response, sends the DLL_SMS_ENQ message at Tm5 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, B.1.3.2
Selection Cond.:	SEL_DLL_SM-TE_STATUS_TX
Preamble:	PRE_CONN_OUTG_INVOKE_INFO-STA
Test description:	FS?DLL_SMS_INFO-STA(PL=?) FS!DLL_SMS_ACK1 START TIMER_Tm5_MINUS START TIMER_Tm5_PLUS ?TIMEOUT TIMER_Tm5_MINUS FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ received after timeout of TIMER_Tm5_MINUS and before timeout of TIMER_Tm5_PLUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_20
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message and received the DLL_SMS_ACK1 message in response, and having sent afterwards a series of DLL_SMS_ENQ messages, each sent at Tm5 expiry from the last acknowledgement received, which are acknowledged with the DLL_SMS_ACK1 message, hangs on after the reception of the DLL_SMS_ACK1 message in response to the 50th DLL_SMS_ENQ message sent.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, 6.3.2.4, table 8
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)

Test description:	FS!DLL_SMS_ACK1
	Set COUNTER = 0
	While COUNTER < 50
	  Set COUNTER = COUNTER + 1
	  START TIMER_Tm5_PLUS
	  START TIMER_Tm5_MINUS
	  ?TIMEOUT TIMER_Tm5_MINUS
	  If COUNTER < 50 Then
	  FS?DLL_SMS_ENQ
	  FS!DLL_SMS_ACK1
	  Endif
	End While
	FS?DLL_SMS_ENQ
	FS!DLL_SMS_ACK1
	FS?DLL_SMS_REL
	Or
	CC?CALL_RELEASE_ind

Pass criteria:	DLL_SMS_REL message or VB connection release received after having sent the DLL_SMS_ACK1 message in response to the 50th DLL_SMS_ENQ message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_21
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, without receiving any response, and afterwards having sent twice the DLL_SMS_ENQ message at Tm1 expiry, without receiving any response, and then the third time the DLL_SMS_ENQ message at Tm1 expiry, which is acknowledged with the DLL_SMS_ACK1 message, sends the DLL_SMS_ENQ message at Tm5 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, 6.3.2.4, table 8
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	START TIMER_Tm1_PLUS FS?DLL_SMS_ENQ START TIMER_Tm1_PLUS FS?DLL_SMS_ENQ START TIMER_Tm1_PLUS FS?DLL_SMS_ENQ FS!DLL_SMS_ACK1 START TIMER_Tm5_PLUS START TIMER_Tm5_MINUS ?TIMEOUT TIMER_Tm5_MINUS FS?DLL_SMS_ENQ
Pass criteria:	First DLL_SMS_ENQ received 3 times, then DLL_SMS_ENQ received a fourth time, in the Tm5 window, after having sent DLL_SMS_ACK1.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_VAL_22
Purpose:	Verify that in case of a TL message larger than 255 octets (three segments), the SM-TE, having received the DLL_SMS_ACK1 message in response to the first DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1", and no DLL message in response to the second DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1", sends the DLL_SMS_ENQ message at Tm1 expiry and after the reception of the following DLL_SMS_ACK1 message, sends again the second DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1".
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, 6.3.2.4, table 8
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=2, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1 FS?DLL_SMS_INFO-MO(E=1, PL=?) START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_ENQ FS!DLL_SMS_ACK1 FS?DLL_SMS_INFO-MO(E=1, PL=?) (see note)
Pass criteria:	DLL_SMS_ENQ received after timeout of TIMER_Tm1_MINUS and before timeout of TIMER_Tm1_PLUS, DLL_SMS_INFO-MO received (retransmitted).
Postamble:	POST_TESTER_RELEASE_VB
NOTE: The payload must be same as the payload received with the last DLL_SMS_INFO-MO message.	

	UBS2_DLL_OUT_DAT_VAL_23
Purpose:	Verify that in case of a TL message larger than 255 octets (three segments), the SM-TE, having received the DLL_SMS_ACK1 message in response to the first DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1", and the DLL_SMS_NACK message in response to the second DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1", sends again the second DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1" and after the reception of the following DLL_SMS_ACK0 message, it sends the third DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="0".
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, 6.3.2.4, table 8
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=2, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1 FS?DLL_SMS_INFO-MO(E=1, PL=?) FS!DLL_SMS_NACK FS?DLL_SMS_INFO-MO(E=1, PL=?) (see note) FS!DLL_SMS_ACK0 FS?DLL_SMS_INFO-MO(E=0, PL=?)
Pass criteria:	The second DLL_SMS_INFO-MO message (E=1) received (retransmitted), the third DLL_SMS_INFO-MO message (E=0) received after DLL_SMS_ACK0.
Postamble:	POST_TESTER_RELEASE_VB
NOTE: The payload must be same as the payload received with the last DLL_SMS_INFO-MO message.	

	UBS2_DLL_OUT_DAT_INV_01
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message and received a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message with wrong checksum in response, sends the DLL_SMS_ENQ message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.3.6
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK1, CS=FALSE) FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_INV_02
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-STA (SMS_SUBMIT) message and received a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message with wrong checksum in response, sends the DLL_SMS_ENQ message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.3.6
Selection Cond.:	SEL_DLL_SM-TE_STATUS_TX
Preamble:	PRE_CONN_OUTG_INVOKE_INFO-STA
Test description:	FS?DLL_SMS_INFO-STA(PL=?) FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK1(TSPX_SMS_SUBMIT_REP_CONF), CS=FALSE) FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ message received in response to a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message with wrong checksum.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_INV_03
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message and received a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message with wrong message length in response, sends the DLL_SMS_ENQ message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.3.6
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(ML= Len(TSPX_SMS_SUBMIT_REP_CONF) + 1, PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_INV_04
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message and received a DLL message with unknown message type in response, sends the DLL_SMS_ENQ message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.3.6
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_UNKNOWN(E=0, PL=TSPX_UNKNOWN_PL) FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_INV_05
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, without receiving any response, and having sent afterwards the DLL_SMS_ENQ message at Tm1 expiry, having received a DLL_SMS_ACK1 message with wrong checksum in response, sends again the DLL_SMS_ENQ message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6 and 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	START TIMER_Tm1_PLUS FS?DLL_SMS_ENQ FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK1, CS=FALSE) FS?DLL_SMS_ENQ
Pass criteria:	DLL_SMS_ENQ message received after DLL_SMS_ACK1 with wrong checksum.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_INV_06
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, without receiving any response, and having sent afterwards a first DLL_SMS_ENQ message at Tm1 expiry, having received the DLL_SMS_ACK1 message with wrong checksum in response, then a second DLL_SMS_ENQ message, having received a DLL_SMS_ACK1 message with wrong checksum in response, and finally a third DLL_SMS_ENQ message, having received a DLL_SMS_ACK1 message with wrong checksum in response, hangs on.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, 6.3.2.4, table 8
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	START TIMER_Tm1_PLUS FS?DLL_SMS_ENQ FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK1, CS=FALSE) FS?DLL_SMS_ENQ FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK1, CS=FALSE) FS?DLL_SMS_ENQ FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK1, CS=FALSE) CC?CALL_RELEASE_ind
Pass criteria:	VB release indication received after 3rd transmission of DLL_SMS_ACK1 message with wrong checksum.
Postamble:	None

	UBS2_DLL_OUT_DAT_INV_07
Purpose:	Verify that in case of a TL message larger than 255 octets (three segments), the SM-TE, having received the DLL_SMS_ACK1 message in response to the first DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1", and a DLL_SMS_ACK0 message with wrong checksum in response to the second DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1", sends the DLL_SMS_ENQ message, and after the reception of the following DLL_SMS_ACK0 message, sends the third DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="0".
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=2, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1 FS?DLL_SMS_INFO-MO(E=1, PL=?) FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK0, CS=FALSE) FS?DLL_SMS_ENQ FS!DLL_SMS_ACK0 FS?DLL_SMS_INFO-MO(E=0, PL=?)
Pass criteria:	DLL_SMS_ENQ message received after DLL_SMS_ACK0 with wrong checksum, 3rd DLL_SMS_INFO-MO message (including the DLL_SMS_INFO-MO message received in the preamble) received with extension bit "0".
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_DAT_INOP_01
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message, without receiving any DLL message in response, and the DLL_SMS_ENQ message at Tm1 expiry, which is acknowledged with a DLL_SMS_ACK0 (SMS_SUBMIT_REP) message, ignores the payload of the DLL_SMS_ACK0 (SMS_SUBMIT_REP) message and sends again the DLL_SMS_INFO-MO (SMS_SUBMIT) message.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	START TIMER_Tm1_PLUS FS?DLL_SMS_ENQ FS!DLL_SMS_ACK0(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_INFO-MO(E=0, PL=?) (see note)
Pass criteria:	Retransmitted DLL_SMS_INFO-MO message received, with the same payload as the DLL_SMS_INFO-MO message received in the preamble.
Postamble:	POST_TESTER_RELEASE_VB
NOTE: The payload received must be the same as the payload received in the preamble.	

	UBS2_DLL_OUT_DAT_INOP_02
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message and received the DLL_SMS_ACK0 message in response, hangs on.
Requirements refs:	6.3.2.1, table 5 and §1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK0 CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after DLL_SMS_ACK0 message.
Postamble:	None

	UBS2_DLL_OUT_DAT_INOP_03
Purpose:	Verify that in case of a TL message larger than 255 octets (three segments), the SM-TE, having sent the first DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1" and received the DLL_SMS_ACK1 message in response, and having sent afterwards the second DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1", and received the DLL_SMS_ACK1 message in response, hangs on.
Requirements refs:	6.3.2.1, table 5 and §1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=2, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1 FS?DLL_SMS_INFO-MO(E=1, PL=?) FS!DLL_SMS_ACK1 CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after the second DLL_SMS_ACK1 message.
Postamble:	None

	UBS2_DLL_OUT_DAT_INOP_04
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_INFO-MO (SMS_SUBMIT) message and received the DLL_SMS_ACK1 message in response, and having sent afterwards the DLL_SMS_ENQ message at Tm5 expiry and received the DLL_SMS_ACK0 message in response, hangs on.
Requirements refs:	6.3.2.1, table 5 and §1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1 START TIMER_Tm5_MINUS START TIMER_Tm5_PLUS ?TIMEOUT TIMER_Tm5_MINUS FS?DLL_SMS_ENQ FS!DLL_SMS_ACK0 CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after DLL_SMS_ACK0 message.
Postamble:	None

7.5 Test purposes for Outgoing call/Release (UBS2_DLL_OUT_REL)

	UBS2_DLL_OUT_REL_VAL_01
Purpose:	Verify that the SM-TE, having received a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message in response to a DLL_SMS_INFO (SMS_SUBMIT) message, sends the DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5, B.1.4.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_VAL_02
Purpose:	Verify that the SM-TE, having received the DLL_SMS_ACK1 message in response to a DLL_SMS_INFO (SMS_SUBMIT) message and a DLL_SMS_ACK1 (SMS_SUBMIT_REP) in response to the following DLL_SMS_ENQ message, sends the DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5, B.1.4.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO_ACK1(LengthInd=0)
Test description:	START_TIMER_Tm5_PLUS FS?DLL_SMS_ENQ FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF).
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_VAL_03
Purpose:	Verify that the SM-TE, having received a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message in response to a DLL_SMS_INFO-STA (SM-TE_STATUS) message, sends the DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5, B.1.4.1
Selection Cond.:	SEL_DLL_SM-TE_STATUS_TX
Preamble:	PRE_CONN_OUTG_INVOKE_INFO-STA
Test description:	FS?DLL_SMS_INFO-STA(PL=?) FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF).
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_VAL_04
Purpose:	Verify that in case of a TL message larger than 255 octets (three segments), the SM-TE, having received the DLL_SMS_ACK1 message in response to a first DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1", the DLL_SMS_ACK0 message in response to a second DLL_SMS_INFO-MO (SMS_SUBMIT) message with extension bit="1", and a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message in response to a third DLL_SMS_INFO (SMS_SUBMIT) message with extension bit="0", sends the DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5, B.1.4.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=2, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1 FS?DLL_SMS_INFO-MO(E=1, PL=?) FS!DLL_SMS_ACK0 FS?DLL_SMS_INFO-MO(E=0, PL=?) FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF).
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_VAL_05
Purpose:	Verify that in the case of two SMs to send, the SM-TE, having received a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message in response to the first DLL_SMS_INFO-MO (SMS_SUBMIT) message and a DLL_SMS_ACK0 (SMS_SUBMIT_REP) message in response to the second DLL_SMS_INFO-MO (SMS_SUBMIT) message, sends the DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	SEL_DLL_MoreSMs_TX
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=2)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_INFO-MO(E=1, PL=?) FS!DLL_SMS_ACK0(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after DLL_SMS_ACK0(PL=TSPX_SMS_SUBMIT_REP_CONF).
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_VAL_06
Purpose:	Verify that, in the case of two SMs to send, the SM-TE, having received a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message in response to the first DLL_SMS_INFO-MO (SMS_SUBMIT) message, a DLL_SMS_ACK0 (SMS_SUBMIT_REP) message in response to the second DLL_SMS_INFO-MO (SMS_SUBMIT) message, and the DLL_SMS_ACK1 message in response to the DLL_SMS_REL message, hangs on.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	SEL_DLL_MoreSMs_TX
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=2)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_INFO-MO(E=0, PL=?) FS!DLL_SMS_ACK0(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!DLL_SMS_ACK1 CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after having sent DLL_SMS_ACK1 in response to DLL_SMS_REL.
Postamble:	None

	UBS2_DLL_OUT_REL_VAL_07
Purpose:	Verify that the SM-TE, having received a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message in response to a DLL_SMS_INFO (SMS_SUBMIT) message and the DLL_SMS_ACK0 message in response to the following DLL_SMS_REL message, hangs on.
Requirements refs:	6.3.2.1, table 5, B.1.4.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!DLL_SMS_ACK0 CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after having sent DLL_SMS_ACK0 in response to DLL_SMS_REL.
Postamble:	None

	UBS2_DLL_OUT_REL_VAL_08
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_REL message, without receiving any DLL message in response, sends again the DLL_SMS_REL message at Tm1 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_REL
Pass criteria:	Second DLL_SMS_REL message received, after timeout of TIMER_Tm1_MINUS and before timeout of TIMER_Tm1_PLUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_VAL_09
Purpose:	Verify that the SM-TE, having sent a first DLL_SMS_REL message, without receiving any DLL message in response, and a second the DLL_SMS_REL message at Tm1 expiry, without receiving any DLL message in response, sends again the DLL_SMS_REL message at Tm1 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_REL START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_REL
Pass criteria:	Third DLL_SMS_REL message received, after timeout of TIMER_Tm1_MINUS and before timeout of TIMER_Tm1_PLUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_VAL_10
Purpose:	Verify that the SM-TE, having sent a first DLL_SMS_REL message, without receiving any DLL message in response, a second DLL_SMS_REL message at Tm1 expiry, without receiving any DLL message in response, and a third DLL_SMS_REL message at Tm1 expiry, without receiving any DLL message in response, hangs on at Tm1 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_REL START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS FS?DLL_SMS_REL START TIMER_Tm1_MINUS START TIMER_Tm1_PLUS ?TIMEOUT TIMER_Tm1_MINUS CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after having received 3 DLL_SMS_REL messages, and after timeout of TIMER_Tm1_MINUS and before timeout of TIMER_Tm1_PLUS.
Postamble:	None

	UBS2_DLL_OUT_REL_VAL_11
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_REL message and received the DLL_SMS_NACK message in response, sends again the DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.4.2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!DLL_SMS_NACK FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after having sent a DLL_SMS_NACK message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_VAL_12
Purpose:	Verify that the SM-TE, having sent a first DLL_SMS_REL message and received the DLL_SMS_NACK message in response, and having sent afterwards a second DLL_SMS_REL message and received the DLL_SMS_NACK message in response, sends again the DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.4.4
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!DLL_SMS_NACK FS?DLL_SMS_REL FS!DLL_SMS_NACK FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after having sent the second DLL_SMS_NACK message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_VAL_13
Purpose:	Verify that the SM-TE, having sent a first DLL_SMS_REL message and received the DLL_SMS_NACK message in response, having sent afterwards a second DLL_SMS_REL message and received the DLL_SMS_NACK message in response, and finally a third DLL_SMS_REL message and received the DLL_SMS_NACK message in response, hangs on.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.4.4
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!DLL_SMS_NACK FS?DLL_SMS_REL FS!DLL_SMS_NACK FS?DLL_SMS_REL FS!DLL_SMS_NACK CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after having sent the third DLL_SMS_NACK message.
Postamble:	None

	UBS2_DLL_OUT_REL_INV_01
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_REL message and received a DLL_SMS_ACK0 message with wrong checksum in response, sends again the DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.4.3
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK0, CS=FALSE) FS?DLL_SMS_REL
Pass criteria:	Second DLL_SMS_REL message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_INV_02
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_REL message and received a DLL_SMS_ACK0 message with wrong DLL message length in response, sends again the DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.4.3
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!DLL_SMS_ACK0(ML=1) FS?DLL_SMS_REL
Pass criteria:	Second DLL_SMS_REL message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_INV_03
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_REL message and received a DLL message with unknown message type in response, sends again the DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.4.3
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!DLL_SMS_UNKNOWN(E=0, PL=TSPX_UNKNOWN_PL) FS?DLL_SMS_REL
Pass criteria:	Second DLL_SMS_REL message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_INV_04
Purpose:	Verify that the SM-TE, having sent a first DLL_SMS_REL message and received a DLL_SMS_ACK0 message with wrong checksum in response, and having sent afterwards a second DLL_SMS_REL message, and received a DLL_SMS_ACK0 message with wrong checksum in response, sends again the DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.4.5
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK0, CS=FALSE) FS?DLL_SMS_REL FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK0, CS=FALSE) FS?DLL_SMS_REL
Pass criteria:	Third DLL_SMS_REL message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_OUT_REL_INV_05
Purpose:	Verify that the SM-TE, having sent a first DLL_SMS_REL message and received a DLL_SMS_ACK0 message with wrong checksum in response, having sent afterwards a second DLL_SMS_REL message and received a DLL_SMS_ACK0 message with wrong checksum in response, and finally a third DLL_SMS_REL message, having received a DLL_SMS_ACK0 message with wrong checksum in response, hangs on.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7, B.1.4.5
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK0, CS=FALSE) FS?DLL_SMS_REL FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK0, CS=FALSE) FS?DLL_SMS_REL FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ACK0, CS=FALSE) CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after having transmitted the third DLL_SMS_ACK0 message with a wrong checksum.
Postamble:	None

	UBS2_DLL_OUT_REL_INV_06
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_REL message and received a DLL_SMS_ACK0 (SMS_SUBMIT_REP) message in response, hangs on (ignoring the payload of the DLL_SMS_ACK0 (SMS_SUBMIT_REP) message).
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!DLL_SMS_ACK0(PL=TSPX_SMS_SUBMIT_REP_CONF) CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after DLL_SMS_ACK0(PL=TSPX_SMS_SUBMIT_REP_CONF).
Postamble:	None

	UBS2_DLL_OUT_REL_INOP_01
Purpose:	Verify that the SM-TE, having received a DLL_SMS_ACK1 (SMS_SUBMIT_REP) message in response to a DLL_SMS_INFO (SMS_SUBMIT) message and the DLL_SMS_ACK1 message in response to the following DLL_SMS_REL message, hangs on.
Requirements refs:	6.3.2.1, table 5, B.1.4.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_INFO-MO(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK1(PL=TSPX_SMS_SUBMIT_REP_CONF) FS?DLL_SMS_REL FS!DLL_SMS_ACK1 CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after having sent DLL_SMS_ACK1.
Postamble:	None

7.6 Test purposes for Incoming call/Establishment (UBS2_DLL_INC_EST)

	UBS2_DLL_INC_EST_VAL_01
Purpose:	Verify that the SM-TE sends the DLL_SMS_EST message or the DLL_SMS_ACK0 (SM-TE_CAPABILITY) message as first DLL message after that the VB connection has been established.
Requirements refs:	6.3.2.1, table 5, B.1.2.1
Selection Cond.:	
Preamble:	PRE_CONN_INC
Test description:	FS?DLL_SMS_EST Or FS?DLL_SMS_ACK0(PL=?)
Pass criteria:	DLL_SMS_EST message or DLL_SMS_ACK0 with payload received.
Postamble:	POST_TESTER_RELEASE_VB

7.7 Test purposes for Incoming call/Data transfer (UBS2_DLL_INC_DAT)

	UBS2_DLL_INC_DAT_VAL_01
Purpose:	Verify that the SM-TE, having established the DL and not received afterwards any DLL message, hangs on at Tm2 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, B.1.3.11 and B.1.4.6
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	START TIMER_Tm2_MINUS START TIMER_Tm2_PLUS ?TIMEOUT TIMER_Tm2_MINUS CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after timeout of TIMER_Tm2_MINUS and before timeout of TIMER_Tm2_PLUS.
Postamble:	None

	UBS2_DLL_INC_DAT_VAL_02
Purpose:	Verify that the SM-TE, having established the DL and receiving afterwards the DLL_SMS_ENQ message, sends the DLL_SMS_ACK0 message in response.
Requirements refs:	6.3.2.1, table 5, B.1.3.7
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!DLL_SMS_ENQ FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 message received after DLL_SMS_ENQ.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_03
Purpose:	Verify that the SM-TE, having established the DL, and having received afterwards the DLL_SMS_INFO-MT (SMS_DELIVERY) message, sends in response the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message before Tm6 or the DLL_SMS_ACK1 message at Tm6 expiry.
Requirements refs:	6.3.2.1, table 5, B.1.3.1 and B.1.3.2
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=0, TSPX_SMS_DELIVER_S1)

Test description:	START TIMER_Tm6_MINUS
	START TIMER_Tm6_PLUS
	Either:
	  FS?DLL_SMS_ACK1(PL=?)
	Or
	  ?TIMEOUT TIMER_Tm6_MINUS
	  FS?DLL_SMS_ACK1(PL=?)
	  Or
	  FS?DLL_SMS_ACK1
	End either

Pass criteria:	DLL_SMS_ACK1 message without payload received after timeout of TIMER_Tm6_MINUS and before timeout of TIMER_Tm6_PLUS, or DLL_SMS_ACK1 message with payload received before timeout of TIMER_Tm6_PLUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_04
Purpose:	Verify that the SM-TE, having established the DL, and having received the DLL_SMS_INFO-MT (SMS_STATUS_REP) message afterwards, sends in response the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message before Tm6 or the DLL_SMS_ACK1 message at Tm6 expiry.
Requirements refs:	6.3.2.1, table 5, B.1.3.1 and B.1.3.2
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=0, TSPX_SMS_STATUS_REP)
Test description:	START TIMER_Tm6_MINUS START TIMER_Tm6_PLUS Either: FS?DLL_SMS_ACK1(PL=?) Or ?TIMEOUT TIMER_Tm6_MINUS FS?DLL_SMS_ACK1(PL=?) Or FS?DLL_SMS_ACK1 End either
Pass criteria:	Either a DLL_SMS_ACK1 message received with payload before timeout of TIMER_Tm6_PLUS or a DLL_SMS_ACK1 message received without payload after timeout of TIMER_Tm6_MINUS and before timeout of TIMER_Tm6_PLUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_05
Purpose:	Verify that the SM-TE, in case it sent a DLL_SMS_ACK1 (SMS_DELIVERY_REP) message in response to a DLL_SMS_INFO-MT (SMS_DELIVERY) message, when receives afterwards the DLL_SMS_ENQ message, sends again the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message previously sent, or, in case it sent the DLL_SMS_ACK1 message in response to the DLL_SMS_INFO-MT (SMS_DELIVERY) message, when receives afterwards the DLL_SMS_ENQ message, sends the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message, if the TL is available, or the DLL_SMS_ACK1 message. In case the terminal sent twice the DLL_SMS_ACK1 message, verify that it sends the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message (if the TL message is available) or the DLL_SMS_ACK1 message after receiving a new DLL_SMS_ENQ message.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=0, TSPX_SMS_DELIVER_S1)
Test description:	START TIMER_Tm6_PLUS Either: FS?DLL_SMS_ACK1(PL=?) FS!DLL_SMS_ENQ FS?DLL_SMS_ACK1(PL=?) or FS?DLL_SMS_ACK1 START TIMER_Tm5_MINUS ?TIMEOUT TIMER_Tm5_MINUS FS!DLL_SMS_ENQ FS?DLL_SMS_ACK1 or FS?DLL_SMS_ACK1(PL=?) End either
Pass criteria:	Either DLL_SMS_ACK1 message with payload received after DLL_SMS_ENQ, or DLL_SMS_ACK1 message without payload received after DLL_SMS_ENQ, but only if the DLL_SMS_ACK1 message received before did also contain no payload.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_06
Purpose:	Verify that the SM-TE acknowledges a first received DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message with the DLL_SMS_ACK1 message, a second DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message with the DLL_SMS_ACK0 message and a third DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="0" message with the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message before Tm6 or the DLL_SMS_ACK1 message at Tm6 expiry.
Requirements refs:	6.3.2.1, table 5, B.1.3.4
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=1, TSPX_SMS_DELIVER_S3)
Test description:	FS?DLL_SMS_ACK1 FS!DLL_SMS_INFO-MT(E=1, PL=TSPX_SMS_DELIVER_S4) FS?DLL_SMS_ACK0 FS!DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S6) START TIMER_Tm6_MINUS START TIMER_Tm6_PLUS ?TIMEOUT TIMER_Tm6_MINUS Either: FS?DLL_SMS_ACK1(PL=?) or FS?DLL_SMS_ACK1
Pass criteria:	DLL_SMS_ACK1 message without payload received in response to the DLL_SMS_INFO-MT message with extension bit "0" after timeout of TIMER_Tm6_MINUS and before timeout of TIMER_Tm6_PLUS, or DLL_SMS_ACK1 message with payload received in response to the DLL_SMS_INFO-MT message with extension bit "0" before timeout of TIMER_Tm6_PLUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_07
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_ACK1 message in response to a first DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message and receiving afterwards the DLL_SMS_ENQ message, sends again the DLL_SMS_ACK1 message.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=1, TSPX_SMS_DELIVER_S3)
Test description:	FS?DLL_SMS_ACK1 FS!DLL_SMS_ENQ FS?DLL_SMS_ACK1
Pass criteria:	DLL_SMS_ACK1 received upon DLL_SMS_ENQ.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_08
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_ACK1 message in response to a first DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message and again the DLL_SMS_ACK1 message previously sent in response to a following DLL_SMS_ENQ message, sends again the DLL_SMS_ACK1 message after having received a new DLL_SMS_ENQ message.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=1, TSPX_SMS_DELIVER_S3)
Test description:	FS?DLL_SMS_ACK1 FS!DLL_SMS_ENQ FS?DLL_SMS_ACK1 FS!DLL_SMS_ENQ FS?DLL_SMS_ACK1
Pass criteria:	DLL_SMS_ACK1 received 3 times.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_09
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_ACK1 message in response to a first DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message and the DLL_SMS_ACK0 message in response to a second DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message and receiving afterwards the DLL_SMS_ENQ message, sends again the DLL_SMS_ACK0 message.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=1, TSPX_SMS_DELIVER_S3)
Test description:	FS?DLL_SMS_ACK1 FS!DLL_SMS_INFO-MT(E=1, PL=TSPX_SMS_DELIVER_S4) FS?DLL_SMS_ACK0 FS!DLL_SMS_ENQ FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 message received after DLL_SMS_ENQ message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_10
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_ACK1 message in response to a first DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message, the DLL_SMS_ACK0 message in response to a second DLL_SMS_INFO-MT (SMS_DELIVERY) with extension bit="1" message and again the DLL_SMS_ACK0 message previously sent to a following DLL_SMS_ENQ message, sends again the DLL_SMS_ACK0 message having received a new DLL_SMS_ENQ message.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=1, TSPX_SMS_DELIVER_S3)
Test description:	FS?DLL_SMS_ACK1 FS!DLL_SMS_INFO-MT(E=1, PL=TSPX_SMS_DELIVER_S4) FS?DLL_SMS_ACK0 FS!DLL_SMS_ENQ FS?DLL_SMS_ACK0 FS!DLL_SMS_ENQ FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 received in response to each DLL_SMS_ENQ message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_11
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_ACK1 message in response to a first DLL_SMS_INFO-MT (SMS_DELIVERY) message with extension bit="1", the DLL_SMS_ACK0 message in response to a second DLL_SMS_INFO-MT (SMS_DELIVERY) message with extension bit="1" and the DLL_SMS_ACK1 message or the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message in response to a third DLL_SMS_INFO-MT (SMS_DELIVERY) message with extension bit="0" and having received afterwards the DLL_SMS_ENQ message, sends again the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message if this was the message previously sent or, in case the message previously sent was the DLL_SMS_ACK1, sends the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message (if the TL message is available) or the DLL_SMS_ACK1 message.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=1, TSPX_SMS_DELIVER_S3)
Test description:	FS?DLL_SMS_ACK1 FS!DLL_SMS_INFO-MT(E=1, PL=TSPX_SMS_DELIVER_S4) FS?DLL_SMS_ACK0 FS!DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S6) Either: FS?DLL_SMS_ACK1(PL=?) FS!DLL_SMS_ENQ FS?DLL_SMS_ACK1(PL=?) or FS?DLL_SMS_ACK1 START TIMER_Tm5_MINUS ?TIMEOUT TIMER_Tm5_MINUS FS!DLL_SMS_ENQ FS?DLL_SMS_ACK1 or FS?DLL_SMS_ACK1(PL=?) End either
Pass criteria:	Either DLL_SMS_ACK1 message with payload received after DLL_SMS_ENQ, or DLL_SMS_ACK1 message without payload received after DLL_SMS_ENQ, but only if the DLL_SMS_ACK1 message received before did also contain no payload.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_12
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_NACK message in response to a DLL_SMS_INFO-MT (SMS_DELIVERY) message with wrong checksum and receiving afterwards the DLL_SMS_ENQ message, sends the DLL_SMS_ACK0 message.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S1), CS=FALSE) FS?DLL_SMS_NACK FS!DLL_SMS_ENQ FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 message received after DLL_SMS_ENQ message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_13
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_NACK message in response to a DLL_SMS_INFO-MT (SMS_DELIVERY) message with wrong checksum and the DLL_SMS_ACK0 message in response to a following DLL_SMS_ENQ message, acknowledges the following DLL_SMS_INFO-MT (SMS_DELIVERY) message with the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message before Tm6 or the DLL_SMS_ACK1 message at Tm6 expiry.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S1), CS=FALSE) FS?DLL_SMS_NACK FS!DLL_SMS_ENQ FS?DLL_SMS_ACK0 FS!DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S1) START TIMER_Tm6_MINUS START TIMER_Tm6_PLUS Either: FS?DLL_SMS_ACK1(PL=?) Or ?TIMEOUT TIMER_Tm6_MINUS FS?DLL_SMS_ACK1(PL=?) Or FS?DLL_SMS_ACK1 End either
Pass criteria:	Either DLL_SMS_ACK1 received with payload before timeout of TIMER_Tm6_PLUS, or DLL_SMS_ACK1 received without payload before timeout of TIMER_Tm6_PLUS and after timeout of TIMER_Tm6_MINUS.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_VAL_14
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message or the DLL_SMS_ACK1 message in response to a DLL_SMS_INFO-MT (SMS_DELIVERY) message, and not receiving afterwards any DLL message, hangs on at Tm2 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=0, TSPX_SMS_DELIVER_S1)
Test description:	FS?DLL_SMS_ACK1(PL=*) START TIMER_Tm2_MINUS START TIMER_Tm2_PLUS ?TIMEOUT TIMER_Tm2_MINUS CC?CALL_RELEASE_ind
Pass criteria:	VB release indication received after timeout of TIMER_Tm2_MINUS and before timeout of TIMER_Tm2_PLUS.
Postamble:	None

	UBS2_DLL_INC_DAT_VAL_15
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message or the DLL_SMS_ACK1 message in response to a DLL_SMS_ENQ message, and not receiving afterwards any DLL message, hangs on at Tm2 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, B.1.3.10
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT(E=0, TSPX_SMS_DELIVER_S1)

Test description:	FS?DLL_SMS_ACK1(PL=*)
	FS!DLL_SMS_ENQ
	If DLL_SMS_ACK1 received with payload
	FS?DLL_SMS_ACK1(PL=?)
	else
	Either
	FS?DLL_SMS_ACK1
	or
	FS?DLL_SMS_ACK1(PL=?)
	End either
	End If
	START TIMER_Tm2_MINUS
	START TIMER_Tm2_PLUS
	?TIMEOUT TIMER_Tm2_MINUS
	CC?CALL_RELEASE_ind

Pass criteria:	VB release indication received after timeout of TIMER_Tm2_MINUS and before timeout of TIMER_Tm2_PLUS.
Postamble:	None

	UBS2_DLL_INC_DAT_VAL_16
Purpose:	Verify that the SM-TE, having established the Data Link Layer connection using the DLL_SMS_ACK0 (SM-TE_CAPABILITY) message and supporting the More Messaging Receiving service, after acknowledging a DLL_SMS_INFO-MT (SMS_DELIVERY) message with the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message or the DLL_SMS_ACK1 message, acknowledges a new DLL_SMS_INFO-MT (SMS_DELIVERY) message with the DLL_SMS_ACK0 (SMS_DELIVERY_REP) or the DLL_SMS_ACK0 message.
Requirements refs:	6.3.2.1, table 5 and §1
Selection Cond.:	SEL_DLL_MoreSMs_RX
Preamble:	PRE_CONN_INC
Test description:	FS?DLL_SMS_ACK0(PL=?) FS!DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S1) FS?DLL_SMS_ACK1(PL=*) FS!DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S2) FS?DLL_SMS_ACK0(PL=*)
Pass criteria:	DLL_SMS_ACK1 message with or without payload received in response to the first DLL_SMS_INFO-MT message and DLL_SMS_ACK0 message with or without payload received in response to the second DLL_SMS_INFO-MT message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_INV_01
Purpose:	Verify that the SM-TE sends the DLL_SMS_NACK message having received a DLL_SMS_INFO-MT (SMS_DELIVERY) message with wrong checksum.
Requirements refs:	6.3.2.1, table 5, B.1.3.5
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S1), CS=FALSE) FS?DLL_SMS_NACK
Pass criteria:	DLL_SMS_NACK received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_INV_02
Purpose:	Verify that the SM-TE sends the DLL_SMS_NACK message having received a DLL_SMS_INFO-MT (SMS_DELIVERY) message with wrong DLL message length.
Requirements refs:	6.3.2.1, table 5, B.1.3.5
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!DLL_SMS_INFO-MT(E=0, ML = Len(TSPX_SMS_DELIVER_S1) + 1, PL=TSPX_SMS_DELIVER_S1) FS?DLL_SMS_NACK
Pass criteria:	DLL_SMS_NACK received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_INV_03
Purpose:	Verify that the SM-TE sends the DLL_SMS_NACK message having received a DLL message with unknown message type.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!DLL_SMS_UNKNOWN FS?DLL_SMS_NACK
Pass criteria:	DLL_SMS_NACK message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_INV_04
Purpose:	Verify that the SM-TE sends the DLL_SMS_NACK message having received a DLL_SMS_INFO-MT (SMS_STATUS_REP) message with wrong checksum.
Requirements refs:	6.3.2.1, table 5, B.1.3.5
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_STATUS_REP), CS=FALSE) FS?DLL_SMS_NACK
Pass criteria:	DLL_SMS_NACK message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_INV_05
Purpose:	Verify that the SM-TE sends the DLL_SMS_NACK message having received a DLL_SMS_ENQ message with wrong checksum.
Requirements refs:	6.3.2.1, table 5
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ENQ, CS=FALSE) FS?DLL_SMS_NACK
Pass criteria:	DLL_SMS_NACK message received.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_DAT_INV_06
Purpose:	Verify that the SM-TE, having acknowledged for three times with the DLL_SMS_NACK message a DLL_SMS_INFO-MT (SMS_DELIVERY) message with wrong checksum, hangs on after the third DLL_SMS_NACK.
Requirements refs:	6.3.2.1, table 5, 6.3.2.2, B.1.3.9
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S1), CS=FALSE) FS?DLL_SMS_NACK FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S1), CS=FALSE) FS?DLL_SMS_NACK FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S1), CS=FALSE) FS?DLL_SMS_NACK CC?CALL_RELEASE_ind
Pass criteria:	VB release indication received after the third DLL_SMS_NACK message.
Postamble:	None

	UBS2_DLL_INC_DAT_INV_07
Purpose:	Verify that the SM-TE, having acknowledged a DLL_SMS_ENQ message with wrong checksum three times using the DLL_SMS_NACK message, hangs on after the third DLL_SMS_NACK.
Requirements refs:	6.3.2.1, table 5, 6.3.2.2
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ENQ, CS=FALSE) FS?DLL_SMS_NACK FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ENQ, CS=FALSE) FS?DLL_SMS_NACK FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_ENQ, CS=FALSE) FS?DLL_SMS_NACK CC?CALL_RELEASE_ind
Pass criteria:	VB release indication received after the third DLL_SMS_NACK message.
Postamble:	None

7.8 Test purposes for Incoming call/Release (UBS2_DLL_INC_REL)

	UBS2_DLL_INC_REL_VAL_01
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message in response to a DLL_SMS_INFO-MT (SMS_DELIVERY) message or having sent the DLL_SMS_ACK1 message in response to the DLL_SMS_INFO-MT (SMS_DELIVERY) message and the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message in response to one of the following DLL_SMS_ENQ messages (when the TL message is available), acknowledges the following DLL_SMS_REL message sending the DLL_SMS_ACK0 message.
Requirements refs:	6.3.2.1, table 5 and §9
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!DLL_SMS_REL FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 message received after DLL_SMS_REL message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_REL_VAL_02
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message in response to a DLL_SMS_INFO-MT (SMS_STATUS_REP) message or having sent the DLL_SMS_ACK1 message in response to the DLL_SMS_INFO-MT (SMS_STATUS_REP) message and the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message in response to one of the following DLL_SMS_ENQ messages (when the TL message is available), acknowledges the following DLL_SMS_REL message sending the DLL_SMS_ACK0 message.
Requirements refs:	6.3.2.1, table 5 and §9
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_STATUS_REP)
Test description:	FS!DLL_SMS_REL FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 message received after DLL_SMS_REL message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_REL_VAL_03
Purpose:	Verify that the SM-TE, after acknowledging a first DLL_SMS_REL message with the DLL_SMS_ACK0 message, acknowledges a second DLL_SMS_REL message sending again the DLL_SMS_ACK0 message.
Requirements refs:	6.3.2.1, table 5 and §16, B.1.4.3
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!DLL_SMS_REL FS?DLL_SMS_ACK0 FS!DLL_SMS_REL FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 message received after the second DLL_SMS_REL message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_REL_VAL_04
Purpose:	Verify that the SM-TE, after acknowledging a first DLL_SMS_REL message with the DLL_SMS_ACK0 message, and a second DLL_SMS_REL message with the DLL_SMS_ACK0 message, acknowledges a third DLL_SMS_REL sending again the DLL_SMS_ACK0 message.
Requirements refs:	6.3.2.1, table 5 and §16, B.1.4.5
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!DLL_SMS_REL FS?DLL_SMS_ACK0 FS!DLL_SMS_REL FS?DLL_SMS_ACK0 FS!DLL_SMS_REL FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 message received after the third DLL_SMS_REL message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_REL_VAL_05
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message in response to a DLL_SMS_INFO-MT (SMS_DELIVERY) message or having sent the DLL_SMS_ACK1 message in response to the DLL_SMS_INFO-MT (SMS_DELIVERY) message and the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message in response to one of the following DLL_SMS_ENQ messages (when the TL message is available), sends the DLL_SMS_NACK message in response to a first DLL_SMS_REL message with wrong checksum and the DLL_SMS_ACK0 message in response to a second DLL_SMS_REL message.
Requirements refs:	6.3.2.1, table 5 and §9, B.1.4.2
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_REL, CS=FALSE) FS?DLL_SMS_NACK FS!DLL_SMS_REL FS?DLL_SMS_ACK0
Pass criteria:	DLL_SMS_ACK0 received as response to DLL_SMS_REL message with correct checksum.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_REL_VAL_06
Purpose:	Verify that the SM-TE, having acknowledged the DLL_SMS_REL message, and not receiving afterwards any DLL message, hangs on at Tm4 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, B.1.4.1, B.1.4.2, B.1.4.3 and B.1.4.5
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!DLL_SMS_REL FS?DLL_SMS_ACK0 START TIMER_Tm4_MINUS START TIMER_Tm4_PLUS ?TIMEOUT TIMER_Tm4_MINUS CC?CALL_RELEASE_ind
Pass criteria:	VB connection release received after timeout of TIMER_Tm4_MINUS and before timeout of TIMER_Tm4_PLUS.
Postamble:	None

	UBS2_DLL_INC_REL_VAL_07
Purpose:	Verify that the SM-TE, having established the Data Link Layer connection using the DLL_SMS_ACK0 (SM-TE_CAPABILITY) message and supporting the More Messaging Receiving service, after acknowledging a DLL_SMS_INFO-MT (SMS_DELIVERY) message with the DLL_SMS_ACK1 (SMS_DELIVERY_REP) message, and a new DLL_SMS_INFO-MT (SMS_DELIVERY) message with the DLL_SMS_ACK0 (SMS_DELIVERY_REP), acknowledges the following DLL_SMS_REL message with the DLL_SMS_ACK1 message.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, B.1.3.10
Selection Cond.:	SEL_DLL_MoreSMs_RX
Preamble:	PRE_CONN_INC
Test description:	FS?DLL_SMS_ACK0(PL=?) FS!DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S1) +RECEIVE_ACK1_WITH_PL FS!DLL_SMS_INFO-MT(E=0, PL=TSPX_SMS_DELIVER_S2) +RECEIVE_ACK0_WITH_PL FS!DLL_SMS_REL FS?DLL_SMS_ACK1
Pass criteria:	DLL_SMS_ACK1 received as response to DLL_SMS_REL message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_REL_INV_01
Purpose:	Verify that the SM-TE sends the DLL_SMS_NACK message in response to a DLL_SMS_REL message with wrong checksum.
Requirements refs:	6.3.2.1, table 5, B.1.4.2.
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_REL, CS=FALSE) FS?DLL_SMS_NACK
Pass criteria:	DLL_SMS_NACK message received as response to DLL_SMS_REL message with wrong checksum.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_REL_INV_02
Purpose:	Verify that the SM-TE sends the DLL_SMS_NACK message in response to a DLL_SMS_REL message with wrong DLL message length.
Requirements refs:	6.3.2.1, table 5, B.1.4.2.
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!DLL_SMS_REL(ML=1) FS?DLL_SMS_NACK
Pass criteria:	DLL_SMS_NACK message received as response to DLL_SMS_REL message with wrong message length.
Postamble:	POST_TESTER_RELEASE_VB

	UBS2_DLL_INC_REL_INV_03
Purpose:	Verify that the SM-TE, having sent the DLL_SMS_NACK message in response to a DLL_SMS_REL message with wrong checksum, and not receiving afterwards any DLL message, hangs on at Tm2 expiry.
Requirements refs:	6.3.2.1, table 5, 6.3.2.3, table 6, B.1.4.4
Selection Cond.:	
Preamble:	PRE_CONN_INC_INFO-MT_RESP(TSPX_SMS_DELIVER_S1)
Test description:	FS!TRANSFER_req(CHS=300, MS=80, PDU=DLL_SMS_REL, CS=FALSE) FS?DLL_SMS_NACK START TIMER_Tm2_MINUS START TIMER_Tm2_PLUS ?TIMEOUT TIMER_Tm2_MINUS CC?CALL_RELEASE_ind
Pass criteria:	VB release indication received after timeout of TIMER_Tm2_MINUS and before timeout of TIMER_Tm2_PLUS.
Postamble:	None

Annex A (informative): Bibliography

ISO/IEC 9646-7: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 7: Implementation Conformance Statements".

History

Document history		
V1.1.1	December 2002	Membership Approval Procedure MV 20030207: 2002-12-10 to 2003-02-07
V1.1.1	February 2003	Publication